Department of Engineering Design and Production

# An Integrated Model-Based Approach for Systems Engineering

**Towards the use of model-based approach in the early development stages of the systems engineering process**

**Sarayut Nonsiri**

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
**DOCTORAL DISSERTATIONS** 19/2015

# An Integrated Model-Based Approach for Systems Engineering

Towards the use of model-based approach in the early development stages of the systems engineering process

**Sarayut Nonsiri**

A doctoral dissertation completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Engineering, at a public examination held at the main building (D) on 12 February 2015 at 1:00 p.m.

**Aalto University**
**School of Engineering**
**Department of Engineering Design and Production**
**Research group on Early development process and Systems Engineering**

**Supervising professor**
Prof. Matti Pietola

**Thesis advisor**
Dr. Eric Coatanea

**Preliminary examiners**
Prof. Yong Zeng, Concordia University, Canada
Prof. Sofiane Achiche, École Polytechnique de Montréal, Canada

**Opponent**
Prof. Eric Blanco, Grenoble Institute of Technology, France

Aalto University publication series
**DOCTORAL DISSERTATIONS** 19/2015



ISBN 978-952-60-6077-4 (printed)
ISBN 978-952-60-6078-1 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-6078-1

Unigrafia Oy
Helsinki 2015

Finland

Publication orders (printed book):
sarayut.nonsiri@aalto.fi

**Author**
Sarayut Nonsiri

**Name of the doctoral dissertation**
An Integrated Model-Based Approach for Systems Engineering

**Publisher** School of Engineering

**Unit** Department of Engineering Design and Production

**Series** Aalto University publication series DOCTORAL DISSERTATIONS 19/2015

**Field of research** Machine Design

**Manuscript submitted** 31 October 2014 **Date of the defence** 12 February 2015

**Permission to publish granted (date)** 23 January 2015 **Language** English

☐ **Monograph** ☒ **Article dissertation (summary + original articles)**

**Abstract**

Systems engineering projects are increasingly complex, and often involve multi-disciplinary teams, different physical locations, and diverse stakeholders. At the early stage of the systems engineering process, systems engineers need to deal with capturing the needs, planning the process, and analyzing early solutions. These activities during early stages have a tremendous impact to all later downstream activities of the development process in term of time-to-market, quality of a system, and final cost.

Model-Based Systems Engineering (MBSE), as a model-centric approach, was purposed to support systems engineers to deal with the complexities in developing a system using models. The advantages of implementing MBSE in a project are for example purposed for improving the quality, reducing the risks, increasing the productivity, and improving the communication effectively. However, models are not very useful if they cannot be executed and used for verification and validation purposes.

The scope of this thesis focuses on integrating a model-based approach formalized by Systems Modeling Language (SysML) in the three following domains: requirements modeling, tasks and activities scheduling, and behavioral analyzing of a physical system.

In the first domain, i.e. requirements modeling, this thesis contributes to the analysis of the impact of changes within requirements with the combination of SysML stereotypes and Design Structure Matrix (DSM) techniques. In the second domain, Design Structure Matrix (DSM) and Differential Evolution algorithm (DDE) techniques allow systems engineers to schedule the development process by minimizing iteration of tasks and increasing concurrency between tasks. In the third domain, techniques from Qualitative Physics, i.e. Dimensional Analysis and Causal ordering, are applied for modeling and simulating the behavior of physical systems.

**Keywords** model-based approach, systems engineering, SysML, design structure matrix, optimization, causal ordering algorithm

**ISBN (printed)** 978-952-60-6077-4 **ISBN (pdf)** 978-952-60-6078-1

**ISSN-L** 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

**Location of publisher** Helsinki **Location of printing** Helsinki **Year** 2015

**Pages** 190 **urn** http://urn.fi/URN:ISBN:978-952-60-6078-1

Aalto-yliopisto

**Tiivistelmä**

**Aalto-yliopisto, PL 11000, 00076 Aalto** www.aalto.fi

**Tekijä**
Sarayut Nonsiri

**Väitöskirjan nimi**
An Integrated Model-Based Approach for Systems Engineering

**Julkaisija** Insinööritieteiden korkeakoulu

**Yksikkö** Koneenrakennustekniikan laitos

**Sarja** Aalto University publication series DOCTORAL DISSERTATIONS 19/2015

**Tutkimusala** Koneensuunnittelu

**Käsikirjoituksen pvm** 31.10.2014 **Väitöspäivä** 12.02.2015

**Julkaisuluvan myöntämispäivä** 23.01.2015 **Kieli** Englanti

☐ **Monografia** ☒ **Yhdistelmäväitöskirja (yhteenveto-osa + erillisartikkelit)**

**Tiivistelmä**

Järjestelmäsuunnitteluprojektit ovat yhä monimutkaisempia ja usein niihin liittyy poikkitieteellisiä tiimejä useissa fyysisissä paikoissa ja erinäisiä sidosryhmiä. Järjestelmäprosessien aikaisessa vaiheessa järjestelmäinsinöörien täytyy kerätä tarpeet, suunnitella prosessi ja analysoida aikaiset ratkaisut. Näillä aikaisen vaiheen toiminnoilla on suuri merkitys myöhempiin kehitysprosessin aktiviteetteihin kuten markkinoille pääsyaikaan, järjestelmän laatuun ja lopulliseen kustannukseen.

Mallipohjainen järjestelmäsuunnittelu (MBSE), mallikeskeisenä menetelmänä, oli tarkoitettu tukemaan järjestelmäinsinöörejä käsittelemään monimutkaisuutta kehitettäessä järjestelmää käyttäen malleja (vai malleja käyttävää järjestelmää?). MBSE:n etuja ovat esimerkiksi laadun paraneminen, riskien vähentäminen, tuottavuuden kasvu ja kommunikaation parantaminen tehokkaasti. Toisaalta mallit eivät ole kovin käyttökelpoisia, jos niitä ei voida toteuttaa tai käyttää validointiin tai verifikaatioon.

Tämän työn keskittyy mallipohjaisen käytännön, mikä on formalisoitu järjestelmänmallinnuskielellä (SysML), integroimiseen seuraavilla aloilla: vaatimusten mallinnus, tehtävien ja toimintojen aikataulutus ja fyysisten järjestelmien käytösanalyysi.

Ensimmäisellä alalla, eli vaatimusten mallinnuksessa, tämä työ edistää vaatimusten muutoksista johtuvaa vaikutusanalyysiä yhdistämällä SysML-stereotyyppi- ja suunnittelun rakennematriisi- (DSM) tekniikoita. Toisella alalla DSM ja differentiaalinen kehitysalgoritmi (DDE) tekniikat mahdollistavat järjestelmäinsinöörien aikatauluttaa kehitysprosessia minimoimalla tehtävien iteroinnin ja lisäämällä tehtävien samanaikaisuutta. Kolmannella alalla kvalitatiivisen fysiikan tekniikoita, kuten dimensioanalyysi ja syy-seuraus analyysi, sovelletaan fyysisten järjestelmien käyttäytymisen mallinnuksessa ja simuloinnissa.

**Avainsanat**

**ISBN (painettu)** 978-952-60-6077-4 **ISBN (pdf)** 978-952-60-6078-1

**ISSN-L** 1799-4934 **ISSN (painettu)** 1799-4934 **ISSN (pdf)** 1799-4942

**Julkaisupaikka** Helsinki **Painopaikka** Helsinki **Vuosi** 2015

**Sivumäärä** 190 **urn** http://urn.fi/URN:ISBN:978-952-60-6078-1

## Tiivistelmä

*Järjestelmäsuunnitteluprojektit ovat yhä monimutkaisempia ja usein niihin liittyy poikkitieteellisiä tiimejä useissa fyysisissä paikoissa ja erinäisiä sidosryhmiä. Järjestelmäprosessien aikaisessa vaiheessa järjestelmäinsinöörien täytyy kerätä tarpeet, suunnitella prosessi ja analysoida aikaiset ratkaisut. Näillä aikaisen vaiheen toiminnoilla on suuri merkitys myöhempiin kehitysprosessin aktiviteetteihin kuten markkinoille pääsyaikaan, järjestelmän laatuun ja lopulliseen kustannukseen.*

*Mallipohjainen järjestelmäsuunnittelu (MBSE), mallikeskeisenä menetelmänä, oli tarkoitettu tukemaan järjestelmäinsinöörejä käsittelemään monimutkaisuutta kehitettäessä järjestelmää käyttäen malleja (vai malleja käyttävää järjestelmää?). MBSE:n etuja ovat esimerkiksi laadun paraneminen, riskien vähentäminen, tuottavuuden kasvu ja kommunikaation parantaminen tehokkaasti. Toisaalta mallit eivät ole kovin käyttökelpoisia, jos niitä ei voida toteuttaa tai käyttää validointiin tai verifikaatioon.*

*Tämän työn keskittyy mallipohjaisen käytännön, mikä on formalisoitu järjestelmänmallinnuskielellä (SysML), integroimiseen seuraavilla aloilla: vaatimusten mallinnus, tehtävien ja toimintojen aikataulutus ja fyysisten järjestelmien käytösanalyysi.*

*Ensimmäisellä alalla, eli vaatimusten mallinnuksessa, tämä työ edistää vaatimusten muutoksista johtuvaa vaikutusanalyysiä yhdistämällä SysML-stereotyyppi- ja suunnittelun rakennematriisi- (DSM) tekniikoita. Toisella alalla DSM ja differentiaalinen kehitysalgoritmi (DDE) tekniikat mahdollistavat järjestelmäinsinöörien aikatauluttaa kehitysprosessia minimoimalla tehtävien iteroinnin ja lisäämällä tehtävien samanaikaisuutta. Kolmannella alalla kvalitatiivisen fysiikan tekniikoita, kuten dimensioanalyysi ja syy-seuraus analyysi, sovelletaan fyysisten järjestelmien käyttäytymisen mallinnuksessa ja simuloinnissa.*

**Abstract**

*Systems engineering projects are increasingly complex, and often involve multi-disciplinary teams, different physical locations, and diverse stakeholders. At the early stage of the systems engineering process, systems engineers need to deal with capturing the needs, planning the process, and analyzing early solutions. These activities during early stages have a tremendous impact to all later downstream activities of the development process in term of time-to-market, quality of a system, and final cost.*

*Model-Based Systems Engineering (MBSE), as a model-centric approach, was purposed to support systems engineers to deal with the complexities in developing a system using models. The advantages of implementing MBSE in a project are for example purposed for improving the quality, reducing the risks, increasing the productivity, and improving the communication effectively. However, models are not very useful if they cannot be executed and used for verification and validation purposes.*

*The scope of this thesis focuses on integrating a model-based approach formalized by Systems Modeling Language (SysML) in the three following domains: requirements modeling, tasks and activities scheduling, and behavioral analyzing of a physical system.*

*In the first domain, i.e. requirements modeling, this thesis contributes to the analysis of the impact of changes within requirements with the combination of SysML stereotypes and Design Structure Matrix (DSM) techniques. In the second domain, Design Structure Matrix (DSM) and Differential Evolution algorithm (DDE) techniques allow systems engineers to schedule the development process by minimizing iteration of tasks and increasing concurrency between tasks. In the third domain, techniques from Qualitative Physics, i.e. Dimensional Analysis and Causal ordering, are applied for modeling and simulating the behavior of physical systems.*

# Preface

This dissertation was carried out in Early Development Process and Systems Engineering research group, Department of Engineering Design and Production, Aalto University School of Engineering. This research work has been funded by Silicom, HybLab (MIDE), MOSES, SIMPRO, and Graduate school.

First of all I want to express my sincere gratitude to my supervisor Prof. Eric Coatanea for giving me this opportunity to pursuit of Ph.D. and also for his instruction, guidance, and supervision throughout my doctoral studies. I particularly would like to acknowledge Prof. Matti Pietola, who is my official supervisor, for his supporting.

I would like also to acknowledge the preliminary examiners Prof. Yong Zeng from Concordia University and Prof. Sofiane Achiche from Ecole Polytechnique De Montreal for their feedbacks and their invaluable comments for improving the quality of my manuscript. Specially, I am very honored to have Prof. Eric Blanco from Grenoble Institute of Technology as my opponent.

Special thank to Francois for supporting and guiding me toward the completion of my manuscript. In addition, I am thankful to all my colleagues in the research group Tanja, Faisal, Siru, Andrea, Galina, Mohamed, Hannele, and Tuomas for giving me encouragement.

Finally, I am grateful to my family for understanding and supporting me in pursuits of the Ph.D.

Espoo, January 23, 2015,

Sarayut Nonsiri

# Contents

# List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

**I** Sarayut Nonsiri, François Christophe, Eric Coatanéa, Faisal Mokammel. A combined Design Structure Matrix (DSM) and Discrete Differential Evolution (DDE) approach for scheduling and organizing system development tasks modelled using SysML. Accepted for publication in *Journal of Integrated Design and Process Science*, 22 pages, IOS Press 2014.

**II** Sarayut Nonsiri, Eric Coatanéa, Mohamed Bakhouya. A Discrete Differential Evolution algorithm for Product Development Scheduling. In *Proceedings of the ASME 2012 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Chicago, Illinois, USA, pp. 1229-1236, DOI: 10.1115/DETC2012-70390, August 2012.

**III** Sarayut Nonsiri, Eric Coatanéa, Mohamed Bakhouya, Faisal Mokammel. Model-Based Approach for Change Propagation Analysis in Requirements. In *2013 IEEE International Systems Conference (SysCon)*, Orlando, FL, USA, pp. 497 - 503, DOI: 10.1109/SysCon.2013.6549928, April 2013.

**IV** Galina Medyna, Sarayut Nonsiri, Eric Coatanéa, and Alain Bernard. Modelling, Evaluation and Simulation during the Early Design Stages: Toward the Development of an Approach Limiting the Need for Specific Knowledge. *Transactions of the SDPS: Journal of Integrated*

*Design and Process Science*, Volume16, number 3, pp. 111-131, DOI: 10.3233/jid-2012-0006, 2012.

**V** Eric Coatanéa, Sarayut Nonsiri, Mohamed Bakhouya, Panu Kiviluoma and Olof Calonius . Early Modelling and Simulation Approach for Fast Evaluation of Early Design Concepts: A System Dynamics Perspective. In *Proceedings of the ASME 2012 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Chicago, Illinois, USA, pp. 97-109, DOI: 10.1115/DETC2012-70652, August 2012.

**VI** Eric Coatanéa, Sarayut Nonsiri, Ricardo Roca, Faisal Mokammel, Juliane Kruck, François Christophe,. Systematic search for design contradictions in systems' architecture: Toward a computer aided analysis. Accepted for publication in *Transactions of the SDPS: Journal of Integrated Design and Process Science*, Volume x, number x, pp. x-x, DOI: x, IOS Press, 23 pages, IOS Press 2015.

# Author's Contribution

### Publication I: "A combined Design Structure Matrix (DSM) and Discrete Differential Evolution (DDE) approach for scheduling and organizing system development tasks modelled using SysML"

This publication was written, implemented, experimented, and the literature review was performed by the author. This work focuses on supporting systems engineers in both management and technical aspects for planning the engineering project where there are several iterations in development process. The contribution of this work consists in representing the tasks that derived from the requirements in Systems Modeling Language (SysML). The tasks are then automatically transformed into design structure matrix (DSM) for modeling the process architecture. A Discrete Different Evolution (DDE) algorithm is then used for automating the sequencing process in order to reduce the iterations, and also increasing the concurrency of the process. The approach provides an automatic approach for optimizing the scheduling process. The method is especially providing added value in the case of projects involving large number of tasks and task interactions.

### Publication II: "A Discrete Differential Evolution algorithm for Product Development Scheduling"

The author wrote this publication, performed the literature survey, gathered the data from the literatures for performing empirical experimental experiments and implemented the computer software in Matlab. This work presented the use of a discrete different evolution (DDE) algorithm for the purpose of sequencing the activities in the case of complex product development processes. The results of the study have demonstrated that

sequencing using DDE and genetic algorithm (GA) are both very competitive result in term of the quality of the obtained solutions. In addition, DDE is a simpler and easier to use approach since the parameters setting in DDE are less difficult than in GA.

### Publication III: "Model-Based Approach for Change Propagation Analysis in Requirements"

In this conference paper, the main contribution has consisted to integrate model-based approach using systems modeling language (SysML) with higher order design structure matrix method for change propagation analysis in requirements. A case study derived from the Eurobot competition has been used to demonstrate the proposed methodology and to validate the approach. This publication was written, implemented, and the literature survey has been performed by the author. The co-authors contributed the valuable comments and suggestions.

### Publication IV: "Modelling, Evaluation and Simulation during the Early Design Stages: Toward the Development of an Approach Limiting the Need for Specific Knowledge"

This journal paper presented an approach developed to evaluate the different design concepts of physical system at the early design stages where quantitative information is lacking. The model and simulation techniques are developed based on a combination of dimensional analysis and graph theory. The simulation approach is implemented in a system dynamics simulation tool. The author has supported the development of the modeling approach and also the implementation inside the system dynamics approach with the system model.

### Publication V: "Early Modelling and Simulation Approach for Fast Evaluation of Early Design Concepts: A System Dynamics Perspective"

The main contribution of this publication is the development of an approach for modeling and simulation in the early stages using the small

amount of quantitative data available at the beginning of the development process. The approach is based on the combined use of three domains of physics and mathematics: qualitative physics, dimensional analysis, and graph-based representation. The case study of an air bearing is used to demonstrate and validate the proposed framework. The author contributed the development of causal ordering algorithm and also the causal propagation of the design objectives in the graph.

### Publication VI: "Systematic search for design contradictions in systems' architecture: Toward a computer aided analysis"

In this publication, the contribution can be described in the following manner. The publication introduces a new approach to systematically discover design contradictions inside the design architecture of a system. The framework is integrating a SysML model of the system to study, which is input in the computer tool. The computer tool generates a graph representation of the system in the form of the variables used to model the system and their interactions. The variables are then clustered in the tool and objectives are given to the performance variables of the system. A propagation algorithm is then used to propagate the objectives inside the graph. Contradictions are found when contradictory objectives appear on a variable of the system. Later TRIZ inventive principles can be used to overcome the design contradictions. The principle can probably be expended to other types of non-physical systems too. The author supports the main author work by implementing the code for creating the computer-aided tool and also by contributing to the algorithms development.

# 1. Introduction

## 1.1 Background

Engineering projects are becoming increasingly complex as a result of their large scale but also because of the increasing spectrum of technologies and constraints that has to be considered. Modern engineering projects often require the involvement of multi-disciplinary teams. In addition, because of the globalization phenomenon, other constraints such as concurrent developments in multiple geographical locations and increasing numbers of stakeholders have to be considered. The discipline of Systems Engineering (SE) has been developed to address this increasing complexity with the consideration of a technical system as a whole. This holistic view of technical systems enables the complexity emerging from the interactions between parts of these systems to be represented. Systems engineering (SE) involves defining the customer's needs and the required functionality early in the development cycle, documenting the requirements, and then proceeding with design synthesis and system validation while considering the complete problem (INCOSE, 2012). A similar logic is proposed in the product development literature (Ulrich and Eppinger, 2011) but with less emphasis on integration and validation.

SE has traditionally been implemented following a document-based approach. Nevertheless, at the early development stage, the information and the engineering data are usually represented in natural language in unstructured text documents. Consequently, it becomes difficult to ensure consistency and reuse aspects. Model-based Systems Engineering (MBSE) has been introduced to address these limitations, and to support systems engineers in both technical and management aspects (INCOSE,

2012). However, even if MBSE is a modeling approach that endeavors to structure and formalize the data and knowledge used in the development process, the MBSE approach offers limited support for transforming the initial unstructured natural language representation into formal models. There is consequently a need for extra types of support. Both MBSE and SE are organized around a process. This process requires the definition of tasks and a certain viewpoint related to the structuring of the development process. Because of the necessity in modern product and system development of minimizing the development time, the scheduling of development tasks can become a very complex process. In addition, as a result of the dynamic and recursive nature of system development activity, changes in system requirements during the development process should lead to the dynamic updating of the task scheduling. Neither SE nor MBSE can deal with this complexity properly without the support of a new form of computer support tool.

A traditional SE process during the conceptual design phase includes system requirements, design specification, system design and analysis, verification, and validation (INCOSE, 2012). The main concept of the MBSE approach is to represent a system and the related documents in formal models and graphical language for the purpose of visualization and communication within the development project. Nevertheless, what are the current approaches to evaluation and verification during the conceptual design phase in MBSE? The answer is a set of elementary tools that are mainly based on the expertise of individual experts. Is it sufficient when the development process is characterized by growing complexity?

The need to support the complexity management of systems engineering problems, specifically in the conceptual design stage, is especially important because of the cascading impacts generated by early decisions in complex system engineering.

The main concern of systems engineers is to ensure that the system being developed satisfies the customer's needs and to communicate about them with project members from different domains (e.g., mechanical engineers, mechatronic engineers, and computer engineers). Such needs are analyzed and expressed in a more formal and specific manner, called requirements. Well-defined requirements can avoid communication prob-

lems resulting from ambiguity, misunderstanding, and incompleteness. However, how can it be ensured in practice that requirements share those quality characteristics?

All these questions represent matters that are still not fully covered by SE or MBSE approaches. The current trends in system developments are increasing the need for more accurate answers to these challenges.

## 1.2 Research Problem

The introduction above divided the need for further development of the SE and MBSE method into two types of research categories; the organizational and scheduling aspects of the development process and the early evaluation tools for development and system engineering. The goal of this section is how to define precisely the research problems that form the scope of the thesis.

The first problem that has been identified is related to the unmanageable complexity of the scheduling problem when projects evolve dynamically and when the overall complexity of projects generates cognitive overloads that are difficult for individuals or groups of humans to manage. From an organizational perspective, at an early stage of the systems engineering process, many activities/tasks have to be scheduled with the objective of reducing the development time. This can mainly be done at an early stage by maximizing concurrency and limiting the number of iterations. As a result of the iterative and recursive nature of the development process (Zeng, 2008), feedback and the number of iterations can proliferate. These contribute to the quality of the development process but are also time-consuming. Although traditional approaches such as PERT (Project Evaluation and Review Technique), CPM (Critical Path Method), and Gantt charts have been developed, are widely used, and provide useful support for planning activities, nevertheless, they are not designed to handle the concurrencies, feedback, and iterations in the systems engineering process.

The first research question that is analyzed in this thesis is:

- How can a scheduling approach for complex projects be developed which is simultaneously able to minimize the number of iterations and maximize the concurrency? What kind of approach can be used for identifying activities or tasks from a set of requirements?

This main research question implies answering a secondary question, which is:

- What kind of approach can be used for identifying the activities/tasks from a set of requirements?

The second research question is related to the capacity to develop tools that really support the evaluation and validation process in systems engineering and in development tasks in general without it being necessary to wait for implementation and integration and validation later in the development process.

In addition, this desire is constrained by the need to move quickly in the development process and not to spend too much time on these initial evaluation phases. Being able to support these initial verifications and evaluation phases well is also important because early decisions have a major impact on the final costs and later generate cascading impacts on most aspects of the system. Existing modeling and simulation methods usually fail to support designers during these stages, as they require too many details and precise quantitative data that cannot be provided. The second research question emerges directly from this need:

- How can the behavior of early system solutions be modeled when the problem is still poorly defined and when quantitative information is either limited in quantity or absent?

This main research question implies answering a research question, which is:

- What system variables should be selected at early stages, and how can the relationships between the system architecture and variables be modeled with limited information?

## 1.3 Aim of the research

As already stated, the aim of this research work is to propose a set of computer-aided approaches dedicated to the early stages of the systems engineering process. The tools should support the organization, analysis, and validation of the early stages of the development of complex systems.

The computer-aided approaches should be able to perform the analysis using a limited set of poorly defined information. The impact of this research is to improve the general productivity of the development process of complex systems by dynamically optimizing the scheduling process and to be able to act quickly on system design at a stage where the impact of decisions is maximal.

## 1.4 Research methods

The research methods followed in this thesis follow a traditional pattern, starting with the state of the art of the existing literature in Systems Engineering, Model-Based Systems Engineering, and Product Development. The aim of this initial examination of the state of the art was to highlight the potential needs and limitations that will require investigation. The research approach followed by the author in the second phase consists of exploring the feasibility of generating a real contribution in the domains where limitations have been identified.

The approach selected to analyze the feasibility first considers that a real contribution implies the development of new forms of computer support. Computer-aided approaches have been selected because the complexity of the problems to be solved also makes it difficult for the cognitive abilities of humans to deal with the level of complexity imposed by modern system developments.

The research approach then explores methods and approaches from areas related to computing science. Another aspect considered in the research method is the need to integrate the research work into a broader framework and tendency, which is the model-based point of view. Modeling languages are developed to promote such types of perspectives and I

consider that SysML is a promising piece of software that can support the development of model-based systems engineering (MBSE) since the models should become more formal. The reason for this is the need to reduce the fuzziness of the models and increase their interpretability.

In this context, special emphasis is placed on the concept of causality and the analysis of its multiple facets from a deterministic perspective.

The present work focuses on a deterministic perspective supporting the analysis of causality in systems because most of the systems studied in the context of this research can be analyzed properly using deterministic models. Nevertheless, the author has also analyzed other perspectives, such as causality inside Bayesian networks, but has decided within the context of this research to limit the analysis to causal principles in problems that can be modeled using the traditional principles of physics and also by considering the architecture of the systems.

A Design Structure Matrix (DSM) or adjacency matrix (in mathematical terms) has been widely considered as an approach in this work to the processing and representation of graphs. Graph representation has been widely used in this research work, as has the whole set of processing methods associated with graphs. Dynamic system modeling and representation have also been used as a concrete approach to the simulation of the systems.

## 1.5 Scope of the research

The primary scope of this thesis focuses on the development of the system at the early design stage of the systems engineering process.As shown in Figure 1.1, there are three areas in the field of systems engineering that relate to one another, including requirements engineering, systems engineering process planning, and modeling and simulation in the early design stage. These three areas are closely related during the early design stage. The decision making is very important at this stage as it can affect all the later stages of the development process.

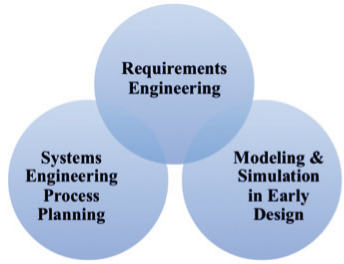


**Figure 1.1.** The scope of this thesis

## 1.6 Author's contribution

As shown in Figure 1.2, the areas where the contributions of this thesis are located in systems engineering are indicated together in Roman numerals.

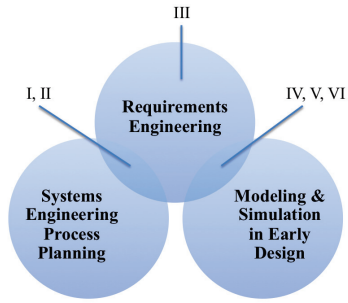


**Figure 1.2.** The contributions of this thesis

## 1.7 Outline of the Thesis

This dissertation consists of six publications. It is organized as follows: Chapter 2 presents the theoretical foundations and related work. The research area combines approaches from different disciplinary backgrounds, including four elements: model-based systems engineering, requirements engineering, process architecture DSM, and dimensional analysis. In Chapter 3 provides a summary of the contribution of this thesis in Publications I, II, III, IV, V, and VI. Finally, the conclusion and perspectives of this dissertation are presented in Chapter 4.

# 2. State of the art

This chapter introduces general background information on systems engineering research fields and related works. First, the bases for the integration of project planning into a model-based approach are presented. This covers model-based systems engineering (MBSE), requirements engineering, Systems Modeling Language (SysML), and computational approaches to sequencing. The second part provides a state-of-the-art study of the field of modeling and simulation during the early stage of the design process and considers qualitative physics, dimensional analysis, and causal analysis.

## 2.1 Systems Engineering (SE)

Engineering projects are increasingly complex and often involve multidisciplinary teams, different physical locations, diverse stakeholders, etc. Most of these projects involve the use of heterogeneous methodologies, tools, and systems models. Since the 1990s, the field of systems engineering has emerged as a favorite for helping engineers cope with such complexity and successfully develop and manage systems and projects that satisfy customer and stakeholder needs while limiting costs, development time, and the use of other resources.

### 2.1.1 Definition of Systems Engineering

A system can broadly be defined as a set of interrelated elements at any level that interact with each other to accomplish a desired goal. A system may include people, data, hardware, software, processes, and facilities. Multiple international standards organizations are currently involved with the development of SE standards, including the International

Council on Systems Engineering (INCOSE), the International Organization for Standardization (ISO), the Institute of Electrical and Electronics Engineers (IEEE), the Electronic Industries Association (EIA), and the Military Standard System Engineering Management (MIL-STD). Each of them has produced a definition of a "system", as follows:

INCOSE: "*A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results.*"(INCOSE, 2012)

ISO 15288: "*An integrated composite that consists of one or more of the processes, hardware, software, facilities, and people that provides a capability to satisfy a stated need or objective.*"(Sheard, 1998)

IEEE 1220: "*A set or arrangement of elements (people, products (hardware and software) and processes (facilities, equipment, material, and procedures) that are related and whose behavior satisfies customer/operational needs, and provides for the life cycle sustainment of products.*"(IEEE-1, 2005)

EIA632: "*An aggregation of end products and enabling products to achieve a given purpose.*"(Electronic Industry Association, 1999)

MIL-STD-499B: "*An integrated composite of people, products, and processes that provide a capability to satisfy a stated need or objective.*"(Sheard, 1998)

The term "systems engineering" was first used in the Bell labs in the 1940s (Schlager, 1956) and, more recently, INCOSE has proposed defining it as "*an interdisciplinary approach and means to enable the realization of successful systems. It focuses on holistically and concurrently understanding stakeholder needs; exploring opportunities; documenting requirements; and synthesizing, verifying, validating, and evolving solutions while considering the complete problem, from system concept exploration through system disposal.*" (INCOSE, 2012). Figure 2.1 illustrates the key elements of systems engineering (Pyster and Olwell, 2013).

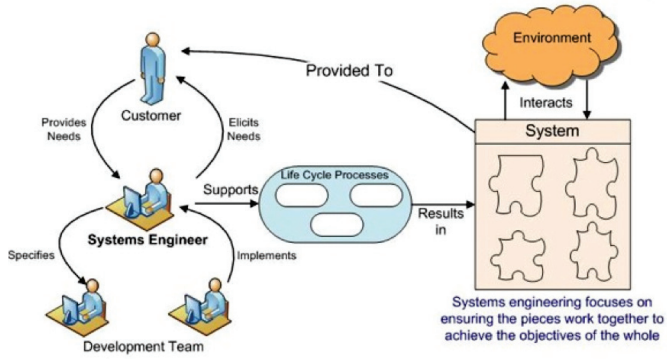


**Figure 2.1.** Key elements of Systems Engineering (Pyster and Olwell, 2013)

### 2.1.2 Model-Based Systems Engineering (MBSE)

*Moving from document-based to model-based approaches*
Engineering data are mostly stored in text documents and these data are used for communication among stakeholders in the different disciplines involved in an engineering project. In recent years, with the progress of computer technology, systems engineering has focused on transitioning from document-based approaches to model-based approaches (Friedenthal et al., 2011). Indeed, traditional document-based approaches can be of limited use when dealing with the complexities of a large project. Document-based approaches often require manual updates and can cause dispersed data, misinterpretation, ambiguity, inconsistency, and unstructured representation of information issues (Kalawsky et al., 2013), leading to project failure. Model-based systems engineering (MBSE) has been introduced to address these issues and to support systems engineers in both the technical and management areas. Several MBSE methodologies have recently been developed and can be found in the literature (Estefan, 2007); they enhance communications, specifications, design, and the reuse of the design and specifications of a system (Friedenthal et al., 2011).

MBSE is a formal modeling approach that captures all the system requirements and the design solutions fulfilling them at different system levels. It supports the systems engineer's activities throughout the systems engineering process (i.e., systems requirements, design, analysis, verification, and validation). The main steps of the systems engineering process are often represented using the "V model" (Figure 2.2) (Forsberg and Mooz, 1999), which is considered an extension of the waterfall model.

This model provides a sequence of steps one can follow during the development life cycle of a project. The process begins with the gathering of the needs from the stakeholders in order to clarify what the system being designed should do. In the next step, the needs are transformed into specific requirements, such as functional, non-functional, and constraint requirements. Next, systems are designed on high and low levels in order to satisfy the different requirements and they are then implemented in both hardware and software systems. The right-hand side of the V shape focuses on the testing, integration, validation, and verification of the subsystems and systems that are designed in each step on the left-hand side of the V. The contents of the V model vary, depending on the application and the project being developed.

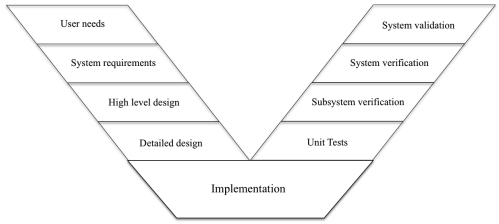


**Figure 2.2.** Systems Engineering Process (V-Model) (Forsberg and Mooz, 1999)

The goal of MBSE is to simplify the complexity of developing and managing complex systems consisting of multiple layers of systems and multidisciplinary teams. The benefits of applying an MBSE approach include enhanced team communication, enhanced analysis, increased reuse, the reduction of development time and risk, early detection of errors, and traceability (Kalawsky et al., 2013) (Holt et al., 2012). Moreover, the application of an MBSE approach does not imply the use of specific tools and platforms. In this work, SysML is used as a modeling language to support the MBSE approach.

### 2.1.3 Systems Modeling Language (SysML)

UML (Unified Modeling Language) (OMG, 2011) is a formal modeling language developed by the Object Management Group (OMG) and is commonly used in software development projects. However, UML is very software-oriented (e.g., object-oriented programming (OOP)) and does not fit into the systems engineering paradigm. Systems Modeling Language (SysML) is a general visual modeling language adapted from UML (OMG,

2012) and is better suited for systems engineering. Figure 2.3 illustrates the similarities and differences between UML and SysML. Mainly, SysML proposes two new diagram types (Requirement and Parametric diagrams); the Use cases, Package, State Machine, and Sequences diagrams are extended from the UML2.0 diagram and the Activity and Block diagrams are reused from UML2.0 with some modifications.

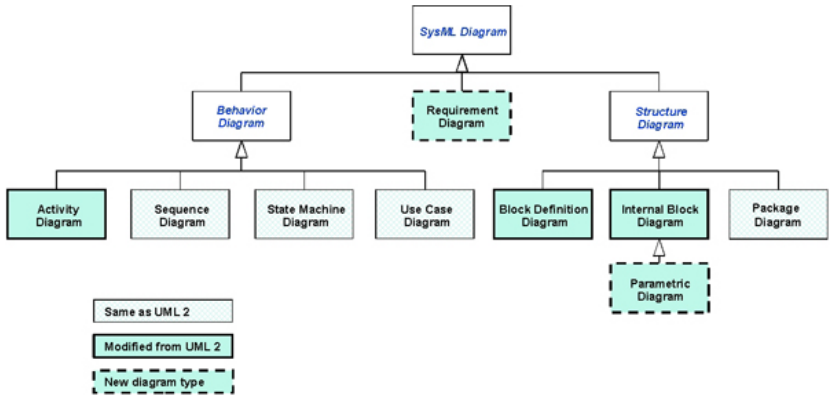


**Figure 2.3.** SysML Diagram types (OMG, 2012)

System models can be executed in SysML and therefore the language can provide support for MBSE. Indeed, SysML allows efficient communication among teams and stakeholders via the different types of diagrams. In addition, the XML Metadata Interchange (XMI) format is used as an interchanging format for exchanging information between computer software tools for enhancing the analysis, such as trade-off studies, impact analysis, and simulation (Holt et al., 2012).

SysML diagrams can be classified into four pillars, as shown in Figure 2.4, based on the type of view, as follows: structure view (block definition and internal block diagrams, and packages), behavior (state-machine, activity and sequence diagrams), requirements (requirements diagram), and parametric (parametric diagram).

Thus, using SysML with MBSE approaches, systems engineering can express requirements and other model elements through models, thus helping to understand, manage, and simplify reality through abstraction (Ramos et al., 2012). In this thesis, it focuses on the requirements diagram, which is significant in the early phases of the design process, and this will be explained in the next section.

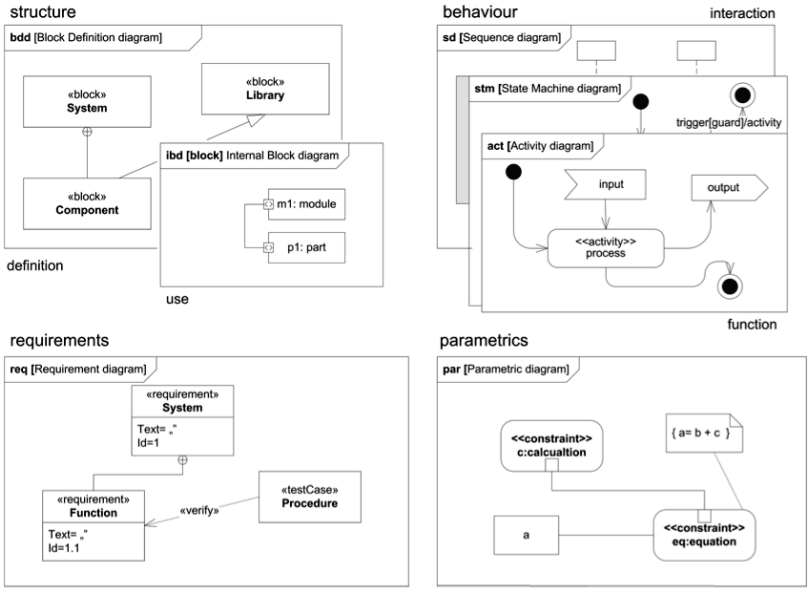


**Figure 2.4.** The four pillars of SysML

## 2.2 Requirements Engineering

Requirements engineering (RE) is an initial step in a product development process. This activity involves discovering the purpose, objective, and needs of a product. During the RE process, stakeholders (e.g., users, clients, developers, and project members) are asked to express and document such needs in a specific way so as to facilitate their communication within the project. The RE process also considers the activities of change management, traceability from solutions back to requirements, and validation and verification of solutions according to requirements. The term 'Engineering' in RE is used to notify the importance of complying with Requirements all through the development process of a product service system (Nuseibeh and Easterbrook, 2000). There are many definitions of RE in the literature. For example, Zave proposes the following definition of RE (Zave, 1997):

"*Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise*

*specifications of software behavior, and to their evolution over time and across software families.* ”

However, Zave's definition mostly focuses on the software engineering domain, with a lack of consideration of other engineering disciplines. Indeed, systems do not work solely through their software parts. RE is considered to be a branch of systems engineering (Stevens et al., 1998). RE is especially prominent at early stages of the systems engineering process. The early stages of systems development have been a major focus of attention for research communities in recent years as these stages have a tremendous impact on the overall outcome of a project. Decisions made during these early phases influence whether the system being developed will meet the needs of the stakeholders.

### 2.2.1 The Requirements Engineering Process and Practices


**Figure 2.5.** Requirements engineering process

The main activities of a RE process considered in the literature are as follows: eliciting requirements, modeling and analysis of requirements, communicating requirements, agreeing requirements, and evolving requirements (Nuseibeh and Easterbrook, 2000), as shown in 2.5.

The first step that is usually considered in a RE process consists of eliciting requirements. This elicitation activity deals with identifying the stakeholders and gathering their needs in order to clarify the design problem and its boundaries. Different techniques exist for eliciting requirements from stakeholders, such as surveys, documents, the manuals of existing systems, prototypes, interviews, and brainstorming. As a generalization, these techniques rely mainly on communication with stakeholders and surveying. As a result, needs are often captured in the form of verbal expressions and sketches.

This phase of RE is relatively difficult but also very important as all the

potential needs should be expressed and reported in a clear, complete, and understandable manner. The difficulty resides in the fact that often stakeholders themselves are not aware of having certain needs or can only express them with difficulty. This is the essence of this task and it is why the term 'eliciting' is used and not only gathering or extracting.

The modeling and analysis of requirements phase mainly consists of having a broad vision of the global organization of requirements vis-à-vis each other and analyzing potential irregularities and mistakes within this set. Requirements are often modeled into different categories corresponding to the specific domain of engineering of the system being developed. In addition, requirements sets are often represented in the form of causal trees for traceability reasons. Each link within such a tree represents a relation between two requirements. The analysis of this model consists of finding the potential inconsistencies between the requirements of this model, e.g., unnecessary redundancies, errors, ambiguities, and contradictions. Such analysis corresponds to the engineering phases of checking for the consistency of the model and the validation and verification phases of the model (V&V).

Communicating requirements is a stage involved with sharing sub-sets of the requirements among the different teams of a project and making sure that all these different teams have understood precisely what is required from them. During this phase, requirements are usually documented and used to communicate between project stakeholders in order to ensure a common understanding of the actual needs of a system. In order to communicate requirements, those requirements should be readable, traceable, and validated. Recently, the researcher has focused on specification languages and notations to support engineers in writing requirements in formal, semi-formal, or natural language so that team members from different domains understand each other. (Bellagamba, 2012)

Agreeing on requirements corresponds to establishing a "contract"between the stakeholders of a project through requirements. This agreement consists of ensuring the acceptance from every part of the definition of stakeholders' needs that will be satisfied during the development of this project. During this stage, a win-win approach can be used to negotiate potential conflicts between stakeholders (Boehm, 1981) by identifying the win con-

ditions of each stakeholder and find a solution that enables mostly winning to occur.

As mentioned at the beginning of this chapter, RE mostly takes place during the early phases of the development process. However, RE also has an important role in the later phases of the development of a system and even when a system is in operation. During these phases of the life cycle of a system, changes in requirements can occur. This stage of RE focuses on the management of change caused by various reasons, such as the emergence of new technology, new needs being introduced, or modifications of existing requirements. Providing well-organized requirements management in a project ensures the possibility of coping with changes but also of reusing requirements for faster development in a future project.

### 2.2.2 Classifications of requirements

In order for the requirements to be modeled, they should be classified into different categories. However, there are many ways to classify them. In this work, the requirements are classified on the basis of their level of detail, which is a common approach in the literature. This way of classification can help the stakeholders to focus on their own concerns without being distracted by unnecessary information. In addition, modeling the requirements on the basis of this requirements classification allows the complexity of the requirements to be simplified by decomposing them into simpler ones until the atomic level is reached. Requirements can mainly be categorized according to their level of precision and at what level they affect a system (*User, System, Sub-system*). Requirements can also be classified into a functional/non-functional decomposition. These two families of classifications are the main ways used for defining categories of requirements.

**Functional/Non-Functional decomposition:**
this classification is used to define what the system should do in terms of actions, and what type of performances it should show.

Functional requirements are often expressed with verbs of action. For example: “The robot shall be capable of handling objects.” In this case, ‘han-

dling objects' defines the function that the system, here a robot, should fulfill. Generally, use cases are used to represent and capture the functions of a system or its components.

Non-functional requirements (NFRs) are also sometimes called quality attributes in software architectures. NFRs are critical elements for stakeholders to define the performances of a system. NFRs can be divided into many categories, such as performance, reliability, robustness, safety, and accuracy.

Non-functional requirements should be expressed in a quantified manner and as precisely as possible. If not, it becomes difficult to evaluate and verify their satisfaction by the system. For example: "The turn-on/-off system of the robot shall exhibit acceptable performance." This requirement is still ambiguous on how to test and verify its satisfaction. Therefore, it should be made precise and often it is refined during elicitation with, for instance, a questioning process of the stakeholder. This requirement could be refined as: "The system should turn off instantly when the off button is pushed. All actuators should remain in their current position and all possible brakes activated." Only then can such a requirement be clearly tested and verified. Requirements might conflict with each other. In such a case, it is of great importance to negotiate between the stakeholders concerned about the real performances that are desired.

**Level of description classification:** For modeling requirements, it is possible to differentiate them with levels of abstraction from the abstract to the more specific requirement level. Traditionally, they are decomposed into four categories: user, system, subsystem, and component requirements (Hull et al., 2005).

In the work of (Soares and Vrancken, 2008), the requirements are classified into two types of requirements: user and system requirements. In the work of (Hull et al., 2005), the requirements are, however, classified into four categories: stakeholder, system, subsystem, and component requirements. It seems that user and stakeholder requirements are basically the same types of requirements since they are involved with the statement of needs on a high level of detail. For the system requirements level in (Soares and Vrancken, 2008), it can be seen as a whole layer that can be

decomposed into system, subsystem, and component (Hull et al., 2005). The requirement classification is represented in Figure 2.6, and the details of each requirement layer are described below.

*User requirements* are on the upper level, which describes the needs of the stakeholders and what they expect from the software/system product being developed. The requirements should not be be a technical description and neither should they provide any solution in advance. Traditionally, requirements of this type are mainly expressed in natural language. The requirements engineers must transform these needs into a formal form as user requirements in order to reduce the ambiguity and misunderstandings. However, this requires communication skills since it involves dealing with stakeholders from different domains of knowledge and with different levels of expertise.

*System requirements* are involved with the system specifications that relate to the software/system architecture to support testing, traceability, and verifying activities. Therefore, it should be clearly stated how the technical aspects required from the system can be testable, traceable, and verifiable. Such requirements are non-functional requirements as they represent different technical criteria of the system (e.g., performance, interface, robustness, security, reliability, maintainability, dimensions, etc.).

As a system is usually broken down into sub-systems following a top-down approach, system requirements are derived at the sub-system level and describe more precisely the requirements related to a specific subsystem. This level of description of the requirements is called *Subsystem requirements*. When the system is broken down further, the expertise of system engineers helps them define components and decide whether these components can be bought, subcontracted, or built in-house. At this level, requirements are called *Components requirements* or configuration items specification. Such requirements represent specific expected performances and constraints for domain-specific components (e.g., mechanical, electrical, software). They stipulate in a formal manner the expected inputs and outputs of each component, as well as the services it provides (description of interfaces) and how it provides them (description of behaviors). When component requirements are described, the subsequent task identification procedure can be performed on the basis of these sets of re-

quirements. The next sub-section presents this procedure in more detail.

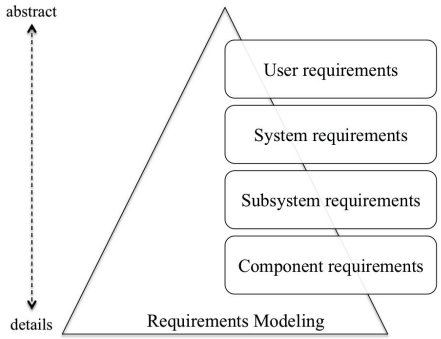


**Figure 2.6.** Requirements levels

For implementing this requirements modeling, a SysML requirements diagram can be used to represent textual requirements and their relationships in a hierarchical order (i.e., user, system, subsystem, and component requirements levels). A requirement is represented as a block, which contains only name, ID, and text information, as shown in Figure 2.7.

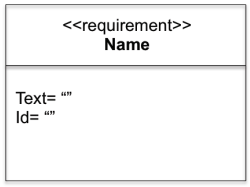


**Figure 2.7.** SysML requirement block

In addition, SysML provides the following set of predefined relationships for representing the relationships among the requirements and other model elements within the system model: derive, satisfy, verify, refine, trace, copy, and containment.

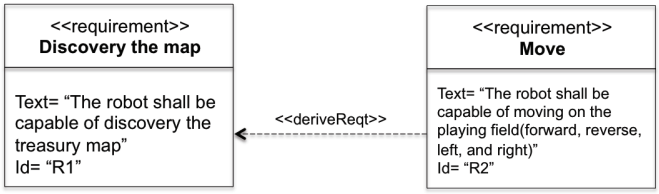


**Figure 2.8.** Derive requirement relationship

- The derive requirement relationship is a dependency representing a requirement generated from another requirement. System engineers cre-

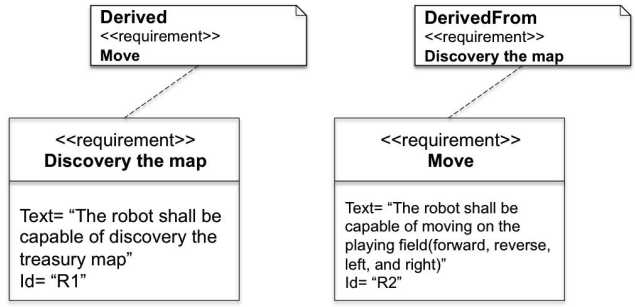


**Figure 2.9.** Derive requirement relationship using callout notation

ate this new requirement during the requirements analysis process if the source requirement needs to be extended into more details. If this relationship between the requirements is not provided, it may cause many problems if any change occurs. The link between the requirements is represented using a dashed line with an open arrowhead. This relationship is applied with the «deriveReqt» stereotype , which is shown at the top of the link. The derived requirement is at the end of the arrow; what is at the front is considered as a source requirement. This kind of dependency is also used in the modeling of requirements on multiple levels, which derive a requirement into more detail. For example, a Move requirement, as the client, is derived from a Discovering the map requirement as the source. A diagram of this requirement is shown in Figure 2.8. However, the SysML requirement relationship can alternatively be represented in two more different ways using callout notation as shown in Figure 2.9. This callout notation in SysML can also be applied to all predefined requirement relationships.

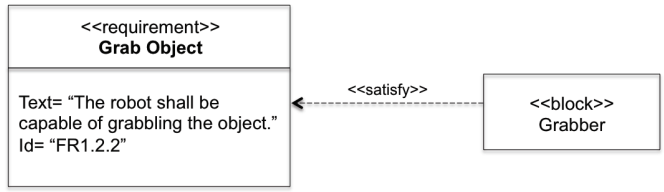


**Figure 2.10.** Satisfy requirement relationship

- The satisfy requirement relationship is the dependency that is applied with the stereotype «satisfy». The fact is that the software/system must be implemented in order to satisfy the requirements. A model element

(e.g., a model, resource, hardware, or software) is designed and allocated to the requirements so as to ensure that the requirements are taken into account. As shown in Figure 2.10, a Grab Object requirement is satisfied by a Grabber block.

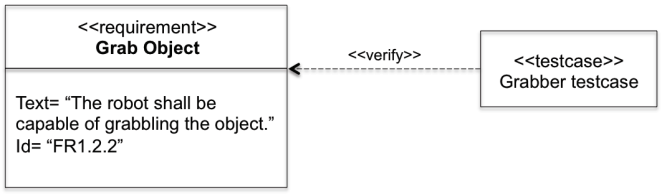


**Figure 2.11.** Verify requirement relationship

- The verify requirement relationship is applied with the stereotype «verify»: this relationship describes the fact that a requirement is verified by the test cases in order to make sure that the requirement is implemented to meet a criterion correctly. For example, a Grab Object requirement is verified by the Grabber test case, as illustrated in Figure 2.11.

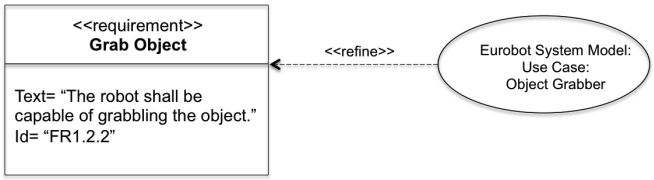


**Figure 2.12.** Refine requirement relationship

- The refine requirement relationship is applied with the stereotype «refine»: this relationship reduces the ambiguity of requirements by specifying an element, i.e., a use case and activity diagram. For example, a Grab Object requirement is refined using a use case to explain in more detail. This is shown in Figure 2.12.

- The standard trace requirement relationship stereotype «trace»: this relationship allows a generic dependency of requirements for traceability between the requirements without any specific detail. As shown in Figure 2.13, a Grab Object requirement is traced by the Grabber system

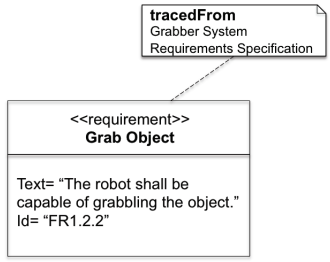


**Figure 2.13.** Trace requirement relationship

requirements specification.

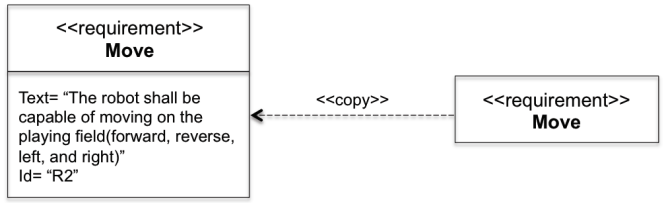


**Figure 2.14.** Copy requirement relationship

- The copy requirement relationship is applied with the stereotype «copy»: this relationship is used for reusing a requirement in another context, as shown in Figure 2.14.

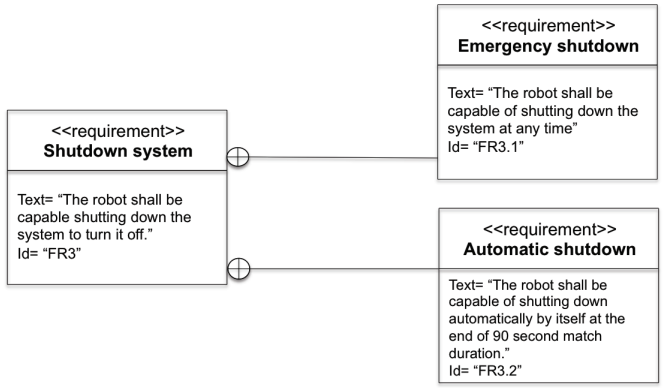


**Figure 2.15.** Containment requirement relationship

- The containment requirement relationship is used for defining a requirement that contains other requirements as a subgroup. This is shown in Figure 2.15.

This predefined set of requirement relationships provide a formal way of modeling the requirements semantically. It also allows the establishment of the traceability of requirements, which is a fundamental capability for checking consistency. In addition, it can be used for impact analysis and change management when the requirements change. Indeed, this can save time and also costs in the development of the product.

## 2.3 Process planning/scheduling and associated domains

### 2.3.1 the description of the phases of design process

A process is viewed by Eppinger as a series of actions or steps that transforms a set of inputs into a set of outputs (Ulrich and Eppinger, 2011). Thus, the inputs and outputs of a process have to be clearly defined for each activity. This definition of a process is very general and need to be associated with a more specific description of the context. A process can be considered from different levels, depending on the viewpoint of the planner, and it can be divided into sub-processes on the basis of the level of detail required. The design process is one of these processes and we will specifically focus on this process in this thesis. The design process is "the organization and management of people and the information they develop in the evolution of a product" (Ullman, 2010). It consists of all the process models of a generic set of phases, including product discovery, project planning, product definition, conceptual design, product development, and product support (Figure 2.16).

The first phase of the design process is product discovery. The product to be developed is identified by gathering the needs from both market and technology sources, such as users, customers, and stakeholders. In another context it can also be imagined that a new technology is directly employed and generates new types of product that have an impact and create new types of needs. This comment was just to show that causality in product development does not always follow the direction from the discovery of customer needs to product development. It can sometimes also be reversed. In the traditional vision, the product discovery can also benefit from knowledge acquired during the exploitation of previous product generations.

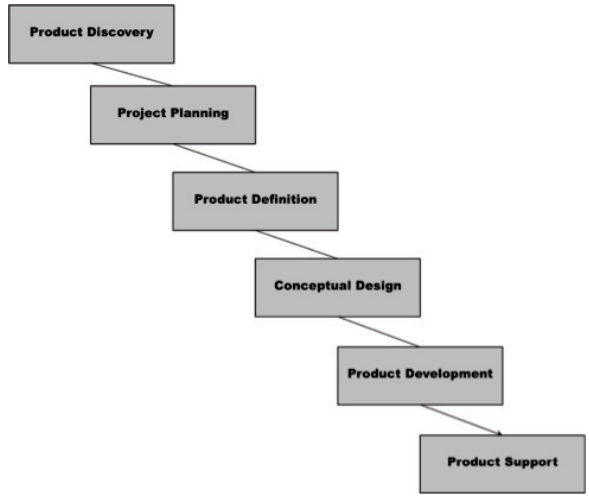


**Figure 2.16.** Design Process (Ullman, 2010)

The second phase is planning the project. The objective of this phase is to identify the set of tasks that need to be accomplished in order to complete the project. Sequencing has an important role in ordering the execution of tasks in an order that reduces resource use.

Product definition is the third phase. It focuses on understanding the problem and decomposing it into smaller problems that are more organizable and manageable. In other words, the goal of this phase is to explicitly clarify the needs into requirements (i.e., function, performance, reliability, safety, etc.). Those requirements reflect the engineering specifications. A quality the specifications should feature is to ensure that ambiguous or incomplete specifications are not provided to the engineering teams and misunderstandings are avoided.

The fourth phase is conceptual design, during which all possible concepts that could fulfill the requirements are generated. All the concepts that are generated satisfy the requirements but only one can be transformed into the final product. Therefore every single concept is evaluated on the basis of multiple criteria and the best solution is selected. The best solution may vary, depending on the objective and also on the resources of the project. The fifth phase is product development and represents the transition from a design concept in the conceptual design phase into the actual product.

Finally, the product support phase includes phases such as maintenance, change management, product-retiring management, and the support provided to the customer.

This initial description has provided an overview of the product development process. This process is constrained by significant time limitations. Indeed, the development time involves costs and potential delays that can harm the future success of the product. For this reason, project planning is an important aspect associated with product development. The next chapter presents this aspect.

### 2.3.2 Project planning process

Project planning is a major issue in every project. It establishes the activities that need to be performed in order to complete the project under cost, quality, and time constraints. As shown in Figure 2.16, the design process is often represented as a set of activities in sequential order. Information flows are provided from the upstream activities to the downstream activities. As modern products are increasingly complex, so is their development. The development team is made up of members from multiple engineering disciplines, who need to communicate and work closely. As a consequence, feedback and iterations are introduced into the process and some features may require several repetitions in order to perform correctly. This can cause delays in the delivery of the project but also make it more difficult to evaluate the time needed to complete the project, especially if the number of repetitions of iterations is unknown. Additionally, there is increasing competition on the global market and companies must minimize the number of iterations in the design process so as to speed up the development of products. A fast decision-making process is also required. Hence, project planning is required to formalize the process to ensure that the outcome of the project achieves the desired goal.

During the building of a project plan information related to tasks should be well documented in order to facilitate communication among the team members who are developing the project. Ullman proposed a plan template that contains the necessary information for each task, such as objective, deliverables, time estimate, cost, and team members (Ullman, 2010). In the project planning process, there are five major steps to be fulfilled:

identification of tasks, definition of the objective, time and resource estimation, task sequencing, and cost estimation (Ullman, 2010).

*Identifying the tasks*

In the proposed process structure, the interactions between tasks are represented by information flow. This information flow may involve data, hardware, software, services, and other resources. In order to have a complete process plan, all the related tasks, as well as their inputs and outputs, must be identified. Identifying a set of tasks initially requires the generation of all the possible planned activities that need to be performed in order to carry out the project. During the creation of the tasks, all the team members must have a good understanding of the problem for the desired goal to be achieved and, thus, the details of the different tasks should be stated as clearly as possible in order to avoid communication problems.

Work Breakdown Structure (WBS) is one of the approaches that can be used to identify the tasks. It provides guidance for decomposing the project into smaller parts (i.e., phases, deliverables, and work packages) in a hierarchical structure which helps to organize and manage the project plan. However, identifying the tasks on the basis of the WBS concept focuses on the parts of the primary systems instead of defining what planned activities need to be performed. In addition, WBS also provides a framework to estimate the cost and resources for the tasks. The tasks themselves can be defined in different manners, depending on the viewpoint and organizational structure of the design team. A deeper study of the different ways that can be considered to define a development task can be found in Section 3.2 of Publication I associated with this thesis.

*Objective definition per task*

After the tasks have been defined, there is a need to concretely state the objective of each task. This statement specifies what is intended to be accomplished in a realistic and achievable period of time. In addition, each task potentially interacts with other tasks via the information flow. It is then necessary to clarify what their outcomes must be for the successor tasks. Additionally, what are the deliverables of the predecessor tasks that can be used for developing a current task? This forms a set of information attributes that flows between tasks. The information flow can

take the form of source code, tests performed, systems model and simulation, analysis results, etc.

The team members should be heavily involved with the process of defining the objectives of the tasks and the nature of the flows coming in and out of the tasks because not knowing what the objectives are can lead to task failure and affect the entire project plan.

*Allocation of people, time, and resources*
This allocation process serves for estimating the time necessary to complete a task and also assign people and other types of resources to the tasks. Depending on the type of organization and the definition of tasks selected, some tasks may need multidisciplinary engineers from the different disciplines to work together on the same task. On the contrary, a specialized team organized according to its domain of competences will also have an impact. Those organizations will have an impact on the internal and external flow of information. The time allocation is estimated in terms of duration and is time-bound. For example, a task requires four months to complete and should be completed before June 15. The definition of the time estimation is a challenging problem and one that is not studied in this research. The common approaches used for time estimation are based on an evaluation of the time complexity of the tasks. This is an area where significant research should be pursued in order to be able to propose a real support tool for task duration. Despite the fact that the scheduling represents only a part of the problem of time fulfillment in product development, this is the focus that has been selected in this thesis. Time and resource allocation have not been studied in this thesis because they were considered to be too company- and organization-specific. Nevertheless, some learning algorithms, such as neural networks, can provide support for developing better time and resource allocation support tools.

In practice, resources will be allocated to the task during the assigned duration and period. A project manager also needs to organize the resources that can be provided when the tasks need them. Consequently, there is also a resource scheduling and allocation process to be implemented. This process is dependent on the scheduling of the development process.

*Task sequencing*

The sequencing of the tasks is a crucial step in project planning. This is the step that is studied in depth in this thesis. It determines the order of the tasks in the process and the succession of tasks. Each task has to provide its result to its successors. In order for the task sequencing to be performed, all the predecessors and successors must already have been identified. The task dependency is sometimes called the information flow. This dependency of tasks can be decomposed into three basic types; serial, parallel, and coupled. Serial tasks are tasks that are performed sequentially in the process and deliver an outcome to successors, which are mostly downstream tasks. Parallel tasks are independent tasks that can be executed simultaneously with other parallel tasks. Coupled tasks are tasks that provide outcomes to one or more other tasks. There are also integrating feedback loops, cycles, and circuits (Eppinger and Browning, 2012). In complex systems development, these type of tasks are commonly present in the process structure.

There are several methods that have been developed for sequencing and scheduling the tasks, such as the Program Evaluation and Review Technique (PERT), Critical Path Method (CPM), and Gantt charts.

*Cost estimation for the project*

The cost estimation of a development project is strongly correlated with the time spent developing the product. The cost structure involves the different project costs. These costs can be decomposed into costs such as the salary costs, equipment costs, prototyping costs, etc.

A correct evaluation of these costs needs to be performed in addition to correct time scheduling in order to manage the project's budget correctly. Different cost elements usually have to be taken into account by managers, such as utilities, equipment, materials, the building, travel, and other resources. This cost estimation of project activities is important because it supports the decision making at management level and to ensure that the necessary funding is available for completing the project. Accurate estimation is needed in order to avoid project failure. However, this topic is not studied in this thesis. Several research studies have been done on the topic of cost estimation and it has been considered that real added value in terms of research effort can be obtained by focusing on the

scheduling process.

### 2.3.3 Practices in project planning

The Critical Path Method (CPM), Program Evaluation and Review Technique (PERT), and Gantt charts are techniques used to support the project management scheduling and organizing activities. CPM/PERT are used to analyze and graphically represent the activities and stages of a project as a network diagram. This network diagram consists of nodes and branches, where a node or circle represents an event or milestone, and an edge represents an activity. Gantt charts use a bar chart to represent the activities from the beginning until the end date.

PERT was developed in the 1950s by the U.S. Navy to support its Polaris nuclear submarine project (Fazar, 1959). It is used to analyze the activities involved and its duration time that need to be developed for completing a given project. The duration time is given by the estimation from the team members, and it is usually represented in terms of elapsed time, rather than actual numbers in units of an hour, week, month, or year. The edge direction indicates the dependency of activities that can be in the form of serial or concurrent activities. PERT is mostly used to schedule large-scale and complex research and development projects.

The expected time is given by calculating the beta probability distribution from equation 2.1.

$$T_e = (T_s + 4 * T_m + T_l)/6 \tag{2.1}$$

where

$T_s$ is the shortest time required for completing an activity, $T_m$ is the most likely time required for completing an activity, $T_l$ is the longest time required for completing an activity.

CPM is a mathematically-based algorithm for sequencing the activities in the project, and also identifying the critical path in a network of activities. It is used to support project management in planning the project. CPM was developed on the basis of network models (Kelley and Walker, 1959). CPM was used to provide support in solving the complexity of task

scheduling in chemical plants. The approach is commonly used in different domains requiring the scheduling of tasks such as product development, software development, and systems engineering. The critical path is the longest sequence of the activities in a network diagram from the start to the end of a project that cannot be delayed. If there is any delay in one of the critical activities, then it will take longer than the deadline to complete the project. In this work, CPM/PERT will be referred to as one single technique.

There are several benefits of using CPM/PERT during the planning phase of a project. The benefits are as follows: (i) CPM/PERT makes it possible to visualize the activities with their relationships; (ii) CPM/PERT makes it possible to present the time necessary to complete each activity and the total project, and also supports the tracking of the progress of an entire project; (iii) CPM/PERT makes it possible to discover which activities are critical and need to be specifically taken care of.

A Gantt chart is a common method used to represent all the activities and the phases of a project in a bar chart. Henry Gantt, an American mechanical engineer, developed this method for project management purposes in the 1910s (Gantt, 1910). This method provides a way of representing the activities that need to be achieved. The Work Breakdown Structure (WBS) can be used to determine those activities. The pre-order and post-order of each activity are defined and links between activities can be drawn. The duration time is also represented. The process is presented as a timeline. Gantt charts are widely used to support represent processes and the advantages of this representation are the following:

- it can be used for tracking and monitoring project schedules;
- it supports the management of time in project management;
- it is used to visualize what the pre-order and post-order activities related to the activity of interest are.

In conclusion, traditional methods such as PERT/CPM and Gantt charts are useful tools used in planning and scheduling engineering tasks. PERT/CPM

is often used for searching for the critical path in a process. Nevertheless, these tools are not designed to design the right level of concurrency and to optimize the number of iterations. They consequently handle a small part of the complexity associated with iterations and concurrency in the systems engineering process. Therefore, this part of the thesis investigates the techniques that can be used to solve the sequencing problem efficiently in order to minimize the number of iterations in the process and to maximize concurrency. In order to fulfill this goal let us now introduce some representation and computing approaches that can manage the complexity associated with the task.

### 2.3.4 Design Structure Matrix (DSM)

A Design Structure Matrix (DSM) is a generic tool that can be used for visualizing, managing, and modeling in complex systems design (Eppinger and Browning, 2012). In the 1960s, Steward introduced a matrix-based technique for solving systems of equations by minimizing the number of iterations in the algorithm, and it was first explained as DSM in (Steward, 1981). In the early 1990s, researchers at MIT utilized and extended this matrix-based technique for applications in research and industry. Currently, DSMs are widely used in a number of applications that can be categorized into four types: product architecture, process architecture, organizational architecture, and multi-domain mapping.

DSMs typically represent elements of a system and their interactions in a square matrix. The relationships between the elements can be represented in binary or numerical form, symbols, or colors, depending on the application. In this thesis, DSMs are mainly considered for their usage in a process architecture for the purpose of sequencing activities/tasks.

*Process Architecture DSM*

Process flow management is extremely useful in project management, as modern companies need to launch new products on the market quickly in order to beat their competitors. Therefore, many researchers have been trying to develop approaches to improving the development process in order to reduce lead-times to market, reduce costs, and increase quality. DSMs have been extended for this type of application (Fernando, 1969) (Hayes, 1969). The scope and purpose of application of DSMs in engineer-

ing is vast. Because of their form, they are also suited to computational implementation. In addition, they can also be applied to any development process involving iterations, regardless of its result. In this application DSMs are often referred to as “process architecture DSM”. However, the terms ‘task-based DSM’, ‘activity-based DSM’, or ‘process DSM’ can also be found in the literature (Eppinger and Browning, 2012)

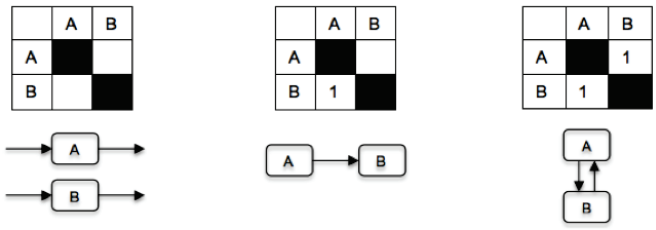


**Figure 2.17.** Three basic types of information flow (a) parallel (b) serial (c) coupled

For visualizing the process architecture in DSM, the task elements and their interactions are represented in a square matrix. The marks in the matrix represent the information flow between the tasks, and also indicate what tasks depend on other tasks or provide information to them. This type of DSM does not need to be a symmetrical matrix. Some tasks do not always send or receive information from other tasks. In this work, the process architecture is represented with the IR/FAD convention, with Input in Row (IR) and Feedback Above Diagonal (FAD). For example, as shown in Figure 2.17 (b), the mark in the second row and first column indicates that activity A provides information to activity B, and therefore activity B depends on activity A. Figure 2.17 (a) (b) and (c) presents the flow of information that can be characterized into the three basic types: parallel, serial, and coupled, respectively.

Project planning in the systems engineering process and in the product development process is demanding. The effectiveness of the process planning can be measured using three major factors, time to market, product cost, and product quality (Ullman, 2010). Iterations in the process are not only a major source of delay but they can also increase project development costs. However, they are often unavoidable as iteration is a typical characteristic of complex processes (Meier et al., 2006). In iteration, a task is performed more than once as a result of feedback or cycles in the process. In the literature, iteration is commonly defined as a repeated activity that can occur throughout the development process and that con-

tinues until the final outcomes are achieved (Ramon Costa, 2003). There are multiple causes of iteration, including results needing to be reviewed and refined, input information not being available when it is needed, the appearance of new information, errors in the results, and changes in requirements. If upstream tasks need to be reworked as a result of iteration, there is a cascade effect on downstream tasks and they will also need to be performed again. Thus, iteration is a major source of possible increases in time and costs linked to the development of a project.

There are many ways to manage iteration, such as: allocating more resources to bottleneck tasks (Yassine and Braha, 2003); improving the flows of information by sequencing the tasks (Yassine et al., 2003); developing new engineering automation tools to execute iterations faster (Yassine et al., 2000). In this thesis, the technique for iteration management is to improve the streamline information flows using DSM and a computational algorithm, which help in ordering the tasks so as to minimize the iteration in the process. DSM can support the computation and visualization of the process and can reduce the complexity of the representation of the process. This can help the planners to realize where the complexities are located. DSM is used in this work together with a computational algorithm used for sequencing the tasks. The next section introduces these approaches.

*Optimization algorithm for task sequencing*

In recent years, several research works have proposed the use of the objective function to sequence tasks (Meier et al., 2006). The main objective of sequencing is to minimize the number of iterations. Nevertheless, there are other important factors that need to be considered besides reducing the number of iterations, such as reducing iteration in a binary matrix (Steward, 1981), reducing iteration in a weight matrix (Kusiak and Wang, 1993), reducing iteration length (Gebala and Eppinger, 1991), reducing iteration and crossover (McCulley and Bloebaum, 1996), and increasing concurrency (Todd, 1997). This work focuses specifically on methods derived from artificial intelligence (AI). Linear programming, which is an approach that has the potential to solve such types of problems, has not been considered in this work because of the nature of the combinatorial problem. AI methods are potentially the only type of methods that can currently be used for the study problem. This is the main reason why ap-

proaches derived from AI have been selected in this work. Nevertheless, all these approaches have a similar pattern, which needs to be described below.

Most algorithms in the literature use two steps to solve the sequencing problem: partitioning and sequencing. Partitioning aims to separate tasks into disjoint groups based on types of information flow (see Figure 2.18). Partitioning separates a DSM into disjoint groups of activities, called coupled blocks, that can be sequenced in parallel. In recent years, several partitioning algorithms have been proposed (Warfield, 1973) (Steward, 1981) (Tarjan, 1972), including the Deep First Search (DFS) algorithm proposed by (Tarjan, 1972). This algorithm is used later in this work. This algorithm helps to find strongly connected components (SCCs) in linear time $O(n+e)$, where n is the number of activities (nodes) and e is the number of relationships (edges). Each SCC makes up a subgraph in which there is a path that can be found from each node to every other node. Figure 2.18 (a) shows information flows before partitioning using DFS in order to find SCCs. Figure 2.18 (b) presents three SCCs that were obtained after applying DFS. In the first group, there is only one task (a). In the second group, there are four detected tasks, which are nodes b, c, d, and e. There is a path to move from one node to every node within this group. In the third group, there are two nodes, which are nodes f and g.

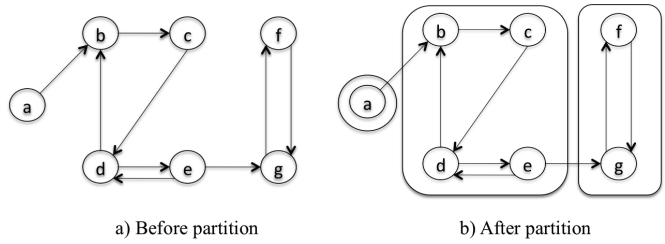


**Figure 2.18.** Strongly Connected Components graph

The sequencing in the process aims to reorder the tasks within coupled blocks in order to minimize the number of feedback relationships between tasks. In Figure 2.19 presents the number of feedbacks/iterations above the diagonal in DSM before sequencing, and illustrate the performance after sequencing in order to minimize the amount of feedback/number of iterations in the process. The complexity of the sequencing problem can be represented as a Quadratic Assignment Problem (QAP), which is an

NP-hard problem. It means that there is no efficient algorithm that can find the optimal solution in polynomial time.

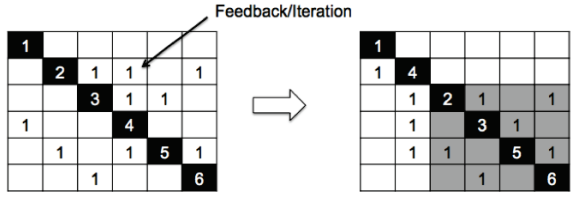


**Figure 2.19.** Sequencing the tasks

In this work, the quality of a process is evaluated using three factors, the number of iterations, iteration length, and concurrency. The first factor is the number of iterations and is a major source of risk for project planning, as mentioned earlier. DSM is used here to visualize the process architecture and gain an understanding of the relationships between tasks. The strength of a relationship does not necessarily need to be binary but can also be coded as a numerical value. It is weighted between 0 and 1 by a domain expert. The strength of a relationship is considered from the amount of information that flows between tasks. This information can be captured from documents, interviewing, meeting, and survey sheets. As illustrated in Figure 2.20, a DSM is a binary matrix with task numbers' IDs indicated in the diagonal cells. There are six iterations present, as indicated by the numbers above the diagonal line. The goal is to minimize them as much as possible by reordering the different tasks, thus making the associated binary numbers move below the diagonal line.

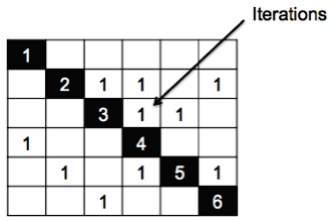


**Figure 2.20.** Iterations in DSM

Iteration length is the second factor that needs to be taken into consideration. As illustrated in Figure 2.21, it presents the cell in the second row and sixth column, where the dependency exists. This cell is an iteration that is at a considerable distance from the diagonal line. It may cause an increase in the cost and time for rework that is much more than the short length of an iteration since it may require all the downstream tasks to be reworked. The idea is that this iteration should be moved below the

diagonal line or minimized by being brought as close to the diagonal line as possible.

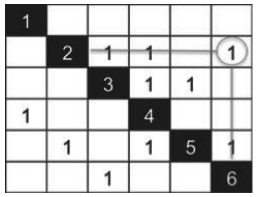


**Figure 2.21.** Iteration length

The third factor is the improvement of the process by increasing the concurrency in order to accelerate the speed of the development project. In Figure 2.22, this is illustrated by the information flow in DSM, which is located close to the diagonal, which can represent the sequential realization of tasks. This sequential process can take time to complete, according to the structure of the process, since the tasks need to wait for predecessor tasks to be executed. Therefore, tasks should be rearranged so as to increase the number of parallel tasks (when possible). The way to achieve this goal is that tasks should be reordered close to the bottom left-hand corner. If the tasks are reordered close to the left-hand side or bottom, then the tasks can be performed simultaneously, as illustrated in Figure 2.23 (a) and Figure 2.24 respectively.

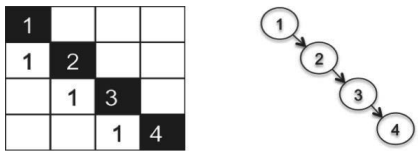


**Figure 2.22.** Sequential tasks

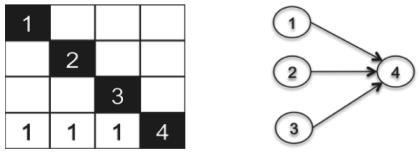


**Figure 2.23.** Parallel tasks

In this work, the objective function in (Scott, 1999) was used to evaluate the quality of the solution because it is not just reducing the number of iterations that counts but also reducing the iteration length and increasing the concurrency of the process by including these three factors in a function as shown in equation 2.3.

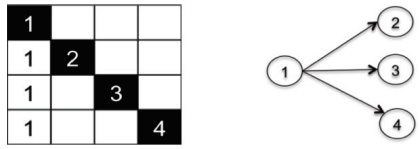


**Figure 2.24.** Parallel tasks

$$f = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \Omega(i,j) * w(i,j) \tag{2.2}$$

where

$$\Omega(i,j) = \begin{cases} 1 * [j + (n-1)]^2 & \text{if} \quad i > j \\ 100 * [j + (n-1)]^2 & \text{if} \quad i < j \end{cases} \tag{2.3}$$

$i$ is the row index, $j$ is the column index, $w(i,j)$ is the strength of the information flow between the tasks $i$ and $j$, $n$ is the total number of tasks in the DSM matrix, and $\Omega(i,j)$ is a weighted distance.

## 2.4 Modeling and simulation in early design stages

### 2.4.1 Dimensional Analysis (DA)

Dimensional Analysis (DA) is a classical method used in the fields of physics, chemistry, engineering, etc. Bridgman provided the following definition of dimensional analysis: "The principal use of dimensional analysis is to deduce from a study of the dimensions of the variables in any physical system certain limitations on the form of any possible relationship between those variables" (Bridgman, 1969). The idea is that dimensional analysis is used to verify a dimensional consistent of variables in a physical equation related to dimensional homogeneity by considering the magnitude and its dimension. This method allows the number of independent variables that are not involved with the physical relationships of the particular problem to be minimized, which can lead to significant reduction of the complexity of a physics problem. Another use of dimensional analysis is that it can also be used as a conversion unit that transforms the measurement of units into different dimensions, for example, transforming a time unit from a day into a second. In addition, dimensional analysis is considered to be a powerful tool to create a small-scale model of physical systems in order to study and experiment the interrelation-

ship of system variables before constructing a prototype in full scale. This is often called the concept of similarity, which refers to two similar things that feature some different elements (Szirtes, 2006) (Sonin, 2001).

There are many physics quantities that appear in physical equations, such as force, time, volume, and area (Sonin, 2001). These quantities can be measured to express a standardized quantity of a physical property in the form of a unit of measurement. It is very useful as a standard to measure the identical physical quantity. For example, time is a physical quantity, and a second is a unit of time for representing a definite predetermined time. According to Newton's laws of motion, there are three units in the following: mass [M], length [L], and time [T] (Butterfield, 2001). These physical quantities are regarded as base units since they are independent of other physical properties. All other units are derived from base units, and are products of the power of base units. They are known as derived units. The general term of dimensional representation can be represented as in equation 2.4, where $\alpha$, $\beta$, and $\gamma$ are constants (Bridgman, 1969):

$$M^{\alpha}.L^{\beta}.T^{\gamma} \tag{2.4}$$

For example, the dimension of speed is meters per second ($ML^{-1}$). Thus, the speed is dependent on both length and time variables. In DeJong's work, dimensional analysis was proposed for application as a tool in economics, and the dollar ($) is regarded as the base unit (de Jong, 1967).

There are three most basic properties, including the principle of dimensional homogeneity, product theorem, and Buckingham's (Pi) theorem (Buckingham, 1914). They are the basis of theory of dimensional analysis (TDA) in order to apply dimensional analysis with different applications.

Principle of dimensional homogeneity: in any physical equation, there are two criteria that must be followed: (a) both sides of the physical equation must be numerically equal; (b) both sides of the physical equation must have dimensional homogeneity.

$$y = \sum a_i f_i(x_i) \tag{2.5}$$

The equation 2.5 represents a physical law, where $a_i f_i(x_i)$ must have the same dimensions as $y$.

For the dimensionality of a physical quantity, it can be either dimensional or dimensionless. A dimensionless number can be represented as a Pi number ($\Pi$). It is used to determine and describe the behavior of a physical model and the interactions of system variables. The Buckingham theorem, also called the $\Pi$ theorem, has shown that a dimensionless number that describes the physical description of a phenomenon can be obtained by reducing the number of variables and combining them together. As a result, it has recently been used to produce several well-known dimensionless numbers such as the Reynolds number, Froude number, and Poiseuille number.

Buckingham's Pi Theorem: It states that a physical equation must be complete as shown in equation 2.6, where $q_1, q_2, ..., q_n$ are the given physical quantities. This theorem is used for computing a dimensionless number from the given physical quantities, even from an unknown equation (Buckingham, 1914).

$$f(q_1, q_2, ..., q_n) = 0 \tag{2.6}$$

The solution can be written as shown in equation 2.7, where $n$ is the number of physical quantities, $r$ is the number of base units, and $\Pi$ is a dimensionless number. The number of a dimensionless number is $n - r$.

$$f(\Pi_1, \Pi_2, ..., \Pi_{n-r}) = 0 \tag{2.7}$$

In equation 2.8 represents the dimensionless parameters $\Pi_i$, where $y_i$ is a performance variable, $x_i$ is a repeating variable, and $\alpha_{ij}|1 \leq i \leq n - r, 1 \leq j \leq r$ are the exponents. Each $\Pi_i$ represents a particular physical situation. The heuristic process for the selection of repeating variables from the given variables can be found in the work of Bhaskar and Nigam (Bhaskar and Nigam, 1990).

$$\Pi_i = y_i.(x_1^{\alpha i1}...x_r^{\alpha ir}) \tag{2.8}$$

### 2.4.2 Qualitative modeling using dimensional analysis

One of the applications of TDA is in the area of Artificial Intelligence (AI) for automating reasoning about the continuous aspects of the physical world in qualitative rather than quantitative information. It is often called Qualitative Reasoning (QR). The main work on QR using dimen-

sional analysis can be found in Kokar's work (Kokar, 1987), and the related work on QR in this field can be found in the literature, such as Qualitative Process Theory (Forbus, 1984) and Qualitative Simulation (Kuipers, 1984).

This thesis focused on the approach of Bhaskar and Nigam (Bhaskar and Nigam, 1990) by using dimensional analysis to reason the behavior of a physical system. In addition, the extended work of Shen and Peng (Shen and Peng, 2006) is also considered in order to generate causal dependencies between system variables and the causal impact in the form of a causal ordering graph. Therefore, these approaches fit into this work, which is involved with the early design stage in the systems engineering process, since they require only a small amount of the information.

# 3. Contributions of this thesis

This section is an attempt to provide a summary and a synthesis of the main contributions of this thesis. Within the focus of developing methodologies to support systems engineers at the early stages of development, two main research questions were raised in Section 1.2 as the scope of this thesis:

- How can a scheduling approach for complex projects be developed which is simultaneously able to minimize the feedbacks and maximize the concurrency? What kind of approach can be used for identifying activities or tasks from a set of requirements?

- How can the behavior of early system solutions be modeled and simulated when the problem is still poorly defined and when quantitative information is limited in amount or absent?

According to these research questions, the main research contributions can be divided into the two following sections:

- Model-based approach for process architecture (Section 3.1): this section deals with the models, methodology, and algorithms proposed in this thesis for assisting engineers in scheduling complex projects with the maximization of concurrency and limitation of feedback iterations. This section refers to the contributions of the author in Publication I, Publication II, and Publication III.

- Modeling and Simulation at early stages of the design process (section

3.2): this section suggests the use of models that facilitate qualitative simulation and the representation of causality and contradiction between design variables. This section refers to the contributions of the author in Publication IV, Publication V, and Publication VI.

### 3.1 Model-based approach for process architecture DSM

The first line of contributions is involved with project planning, where many iterations occur in the process architecture. These iterations are sources of risk that might generate an increase in the project length. The proposed framework integrates a model-based approach modeled using a design structure matrix (DSM) and discrete differential evolution (DDE) to optimize the sequencing of tasks. This framework supports the systems engineer and project manager in organizing and sequencing the tasks in an engineering project. The practical use of the proposed framework is demonstrated by means of the case study of a robot for the Eurobot competition.

The complexity of systems engineering projects has increased during recent years since new systems are integrating new technologies and new constraints, for example, environmental constraints. In fact, the dependency and information flow shared between project members are inevitable. The establishment of an appropriate project plan (e.g., a task schedule) is of critical importance in a systems engineering process as it helps to satisfy the constraints related to the time-to-market, cost-efficiency and quality of the product.

Traditionally, the systems engineering process has been implemented using document-based methods to describe the properties of the tasks, such as their name, objective, predecessors, successors, deliverables, and duration (Ullman, 2002). They are mostly stored in the form of a hard copy or electronic files, and are used as communication tools within the project. However, in a large-scale project, there are many tasks in the development process. Moreover, the tasks can be changed as a result of changes in the requirements. Those changes require time and effort to organize, maintain, and verify the requirements and tasks documents. Therefore,

any incompleteness or error that occurs during these stages can affect all the later stages of the development process and could result in an unexpected increase in project costs (monetary, time, etc.). Therefore, an integrated model-based approach is proposed to reap the benefits of using a model-based approach combined with a DSM in a systematic manner to support systems engineers in task identification and sequencing. The proposed framework consists of four steps (i.e., requirement modeling, task identification, process architecture, and task sequencing) as illustrated in Figure 3.1.

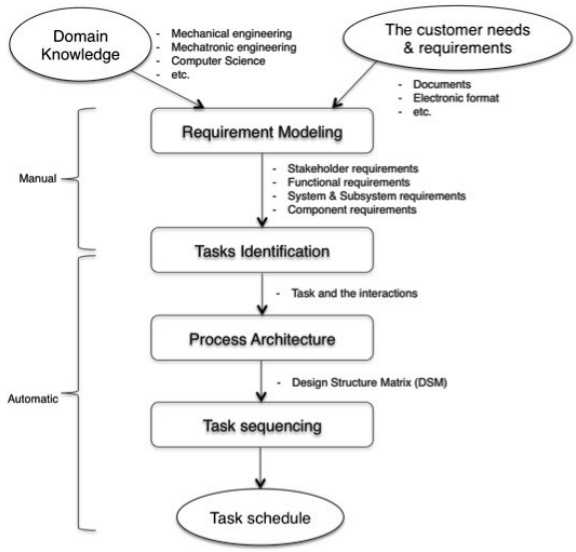


**Figure 3.1.** Integrated Framework

### 3.1.1 Requirement Modeling using SysML

In this thesis, SysML is used to model the requirements since there is a predefined set of requirement relationships for modeling them. Requirements can be modeled on the basis of their levels of detail, from abstract to specific requirements. (Hull et al., 2005) propose four categories: stakeholder requirements, system requirements, subsystem requirements, and component requirements. In this work, a fifth category, functional requirements, has been added 3.2. This category does not overlap with the other categories. Nevertheless, there are dependencies between the layers of requirements. The dependency schema is usually defined as a many-to-many (n-to-n) relationship (Hull et al., 2005). A requirement on a lower level can be derived from many requirements in a higher layer and vice

versa.

The functional requirements usually describe what the system should do. They should not describe how the implementation should be completed. Although the functional requirements can be obtained from stakeholder requirements, the redundancy of a new requirement should be avoided because of the danger of the emergence of the problem of inconsistency. Generally, an actor in the system uses use cases to capture the functionality of the system and its relations to the environment. These requirements will be used to support a systems engineer in analyzing and designing a system. In addition, they are used for testing purposes to verify that these functional requirements are satisfied. The author considers that the advantages of adding functional requirement into the model can be listed as below:

- Communication: since these requirements will be used for communicating with the stakeholders within a project. This requirement level can help the systems engineers focus on their own functions without being distracted by unnecessary information;

- Simplifying the complexity: at the functional requirements level, it allows systems engineers to decompose a complex functional requirement into smaller requirements in order to make the complex functional requirement more manageable;

- Reusability: when developing a new project in the same domain, several functional requirements can be reused by considering the functionality of the existing system. Therefore, adding the functional requirements to a new category can actually save costs and the time needed for eliciting the new requirements.

In Section 4.2 of Publication I, the implementation of modeling the requirements at different levels is presented using SysML.

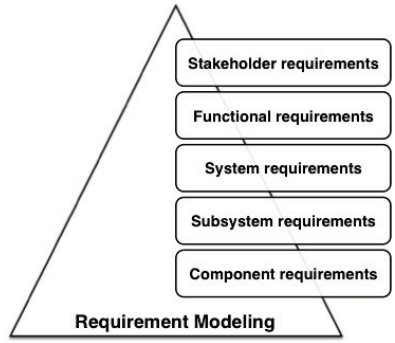


**Figure 3.2.** The requirement levels

### 3.1.2 Identifying Tasks

At this stage, the identification of the tasks is considered. Traditionally, the work breakdown structure (WBS) approach provides a collection of tasks that need to be done. The approach helps in organizing the tasks into work packages classified by domains of activities. The approach supports the decomposition of tasks and activities into smaller chunks (e.g., hardware, software, data, and service).

In this work, a task is derived from requirements. This derivation process should preserve the meaningfulness of development tasks from the engineering and organizational perspectives. Therefore, the tasks should also be identified on the basis of the requirements. This is because a systems engineer is typically given the needs as the input, and then designs the systems that must be developed to satisfy the customer's needs. Thus, any task in the process should also be identified according to the requirement in order to ensure that the requirements are satisfied. The different viewpoints below can be also considered as a development task.

- Requirement category of the system: a development task can be seen as a type of requirement category such as a function, a system, a subsystem, or a component requirement.

- Phase of a development process: a development task can be seen as a phase of the development process such as designing, coding, or testing in the case of software development.

- Function of the functional architecture: a development task can also be seen as a function of a system, for example, the moving module of a mobile robot can be seen as a development task of a mobile robot.

- Subsystem of system architecture: the purpose of this perspective is to consider the physical architecture of a system and to take the subsystems as specific development tasks.

- Domains of technological expertise: this considers a development task as a domain of technological expertise such as mechanical engineering, mechatronics, or computer science.

It is also possible to combine these viewpoints and to imagine combinations between the different viewpoints to form a composite definition of a development task, with the final goal of satisfying the requirements.

One potential solution to the problem of implementing the proposed concept in the existing SysML is to create a simple block named "Task" in a block definition diagram (BDD). Within this block, all the attributes (e.g., ID, name, predecessors, and successors) of a task are included. Therefore, all the tasks and their dependencies could then be represented in the BBD with other components of the system. However, it is a cumbersome and time-consuming process to create each attribute name in order to define its attribute value. In addition, it would be hard for a modeler to differentiate between representations involving objects such as physical components and representations involving tasks.

Another potential solution within SysML is the following. SysML allows a modeler to customize or extend a profile according to their needs. Therefore, for implementing this approach, the author has introduced a new task profile diagram as shown in 3.3. This new stereotype is extended from a Class metaclass that contains the necessary attributes. This class can be seen as a template for defining a description of a task. All the attributes of a task are listed below:

- ID: the ID of a task

- Name: the text of a task.
- Objective: the object of a task.
- Deliverables: what this task will deliver.
- Predecessors: what tasks should be performed before.
- Successors: what tasks should be performed afterwards.
- Start date: the start date of a task.
- Finish date: the finish date of a design task.
- Duration: the duration information of a task.
- Team member: who respond to this task.
- Prepared by: who prepares a task.
- Checked by: who checks a task.
- Approved by: who approves a task.
- Note: the information for a general purpose of a task.

After the tasks and their attributes are given, the relationship «dependency» is used to define the relationship between tasks and requirements. Thus, all the tasks are coupled with requirements and embedded into the SysML model and can be traced. In addition, they can be reused for developing the project plan in a future project.

#### 3.1.3 Transforming the tasks into DSM

This stage transforms the tasks in the SysML model into a Design Structure Matrix (DSM) in order to optimize the sequence of the process. Traditionally, the tasks and their relationships in the form of a DSM are completed manually. If the number of tasks and dependencies are on a

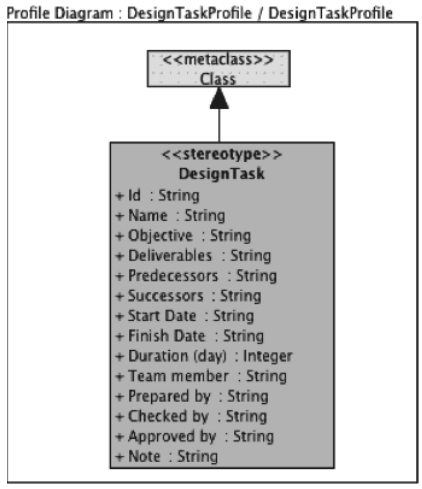


**Figure 3.3.** A task profile diagram

large scale, then it is time-consuming to fill in and to correct the matrix. Incomplete or erroneous input can negatively affect the process planning. In addition, a continuous change of tasks can occur throughout the lifecycle of the system development process. To build the process architecture using the proposed approach, the task attributes defined in the SysML model will be extracted from the XML Metadata Interchange (XMI) code in order to create a task-based DSM that represents the interaction of the tasks. Thus, the DSM matrix is automatically given by the computer software tool and prepared for sequencing the tasks in the process in the next step. In Section 4.4 of Publication I provides an implementation of transforming the tasks into a process architecture DSM.

### 3.1.4 Task sequencing

The process architecture DSM, as given by the previous step, is used to perform the sequencing of the tasks in order to minimize the iterations and increase the concurrency within the process. There are two important steps: partitioning and sequencing. The first step, partitioning, separates the tasks in order to perform the sequencing of all the coupled blocks in parallel. More details are given in Section 2.3 in Publication I. The second step is the sequencing. Traditionally, a Genetic Algorithm (GA) has been applied for sequencing the tasks since it is easy to understand, simple to implement, and provides very good solutions. Nevertheless, it requires many parameters for setting in order to search for the solution by means of a genetic algorithm, such as the crossover probability, mutation prob-

ability, tournament size, and elitism number. These parameters often affect the quality of the results, and it becomes difficult to adjust these parameters in order to obtain satisfactory solutions. Generally, a practical optimization algorithm should find the true global optimum without being affected by the initial parameter values. In addition, it should be self-adaptive and easy to use with a minimum number of parameter settings and should provide fast convergence.

The basic idea of this thesis is that DE was effectively and successfully applied to many numerical optimization problems. In particular, DE simplifies the optimization process by limiting the number of parameters to be adjusted in order to converge toward an optimal solution. In addition, the quality of the solutions obtained with DE has been proven to be very competitive compared to Genetic Algorithms (GA) and most other metaheuristic algorithms, such as Ant Colony and Particle Swarm algorithms. However, DE cannot be applied directly to solving combinatorial problems. In addition, DDE has never been explored before in terms of its application to the problem of the sequencing of tasks. For these reasons, this thesis investigates the utilization of DDE for the optimization of the sequence of tasks in a systems engineering process.

The main difference between the DE and DDE algorithms is that the DE algorithm is applied for solving numerical optimization problems but the DDE algorithm is used for solving combinatorial optimization problems.

GA, as a traditional approach, and DDE are basically the optimization methods which are a subset of a technique called “Evolutionary Computation”. Therefore, the common underlying principle of these algorithms is the same. The main difference between DDE and GA is that GA mimics the natural section process to evolve the population, but DDE uses vector differences to perturb the parent vectors to generate new vectors in the population in the next generation.

Publication II provides details of the implementation of DSM and DDE for the sequencing of tasks in order to reduce the number of iterations and increase the concurrencies in a systems engineering process. There are four instants, in both binary and weighted DSM from previous research, were used to compare the performance between DDE and GA algorithms.

As a result, it has been shown that the quality of the solutions obtained with DDE and GA is very competitive, as shown in Figure 11.

Both DDE and GA are simple in terms of implementation since they only require the fitness function to guide the search mechanism to find the solutions. However, DDE requires two fewer parameters than GA, as shown in Tables 3 and 4 in Publication II, which means it is easier to tune the control parameters in order to optimize the problem. Thus, this approach can support the minimization of the lead time and costs in product development and is easy to use.

## 3.2 Modeling and Simulation at early stages of design process

This second line of contributions focuses on the modeling and simulation of system behavior in the early stages of the design process, where the required knowledge and information are in short supply. Traditionally, existing methods require precise quantitative data, which are usually lacking for supporting engineering designers during these stages. The proposed method is a combination of physics and mathematics: qualitative physics, dimensional analysis, and graph-based representation, in order to quickly model and simulate very early design solutions after the requirements are given.

The contributions that are proposed consist of three algorithms that respectively select the variables of importance in the selection of system variables; defining the causality between the variables of a system and searching for contradictions between the design objectives.

### 3.2.1 System variables selection

One important aspect of system modeling is the selection of the system variables that are involved with a system. This step provides the method that can answer this question: “What are the variables that should be considered when modeling a design concept?” The selection of a set of system variables should rely on a generic classification for modeling a physical system. In order to model the system as a set of components that interact with each other, the input and output of each component

should be defined. The concept of a Bond Graph fulfills this idea since it provides the fundamental notion of considering a system as a set of elementary boxes fulfilling elementary functions and connected to each other by connectors. In addition, it provides the classification of the variables into four different types, as follows: effort, flow, displacement, and momentum variables, as shown in Figure 2 of Publication V. Nevertheless, there is a lack of geometric constraints and other properties such as the properties of materials, fluids, or energies are not captured. Therefore, Coatanea (Coatanéa, 2005) introduced a new type of variable called a connecting variable, which describes the properties of components, materials, and information crossing between components as well as the properties of the black boxes themselves. It can then capture the variables, for example, viscosity, Young's modulus, geometric properties, resistance, and other physical constants, that are involved with the model of a system. In Figure 4 of Publication V presented a set of elementary variables which consists of five different types of variables that are considered in this work as follows: effort (E), flow (F), displacement (d), momentum (m), and connecting variables (C).

### 3.2.2 The extension of Causal ordering algorithm

In this step, this work proposed the extension of the causal ordering algorithm of Shen and Peng in order to model the behavior of a system. The first thing that needs to be considered is the level of detail of the modeling. The idea of this approach is that the system model is designed and represented on a higher level of topology since the information at the early stage is not yet enough to make it possible to decompose such a system into its low-level details. In Figure 6 of Publication V presented a generic topology of the design concept of a pressure regulator. Each box is represented as a component in the system model. The system variables given from the previous step will be properly allocated to the different boxes according to their types. The input for each box is regarded as an exogenous variable. For example, the inlet pressure is an exogenous variable since it is transferred into the pipe within the pressure regulator. Displacement and momentum variables are considered as endogenous variables contained inside the boxes. Connecting variables can affect both the interconnection and the boxes. In Figure 7 of Publication V shows a set of system variables assigned to the boxes of the pressure regulator.

A later step is applying the causal ordering algorithm based on dimensional analysis in order to generate a Pi number for reasoning the system. Section 2.3.2 provides seven prerequisites for applying the heuristic developed by Shen and Peng (Shen and Peng, 2006). As the result, the causal graph of the design concept is obtained for analyzing the causality of the physical system.

### 3.2.3 Contradiction search between design objectives

This contribution is probably the most important of the contributions mentioned since the potential practical impact is important. The ability to find and quickly map design conflicts (or contradictions) in a system is fundamental for a designer. It can provide concrete support for their activities.

The main idea developed in this contribution is first to cluster the system variables into four categories, named the performance variables, the independent design variables, the dependent design variables, and the exogenous system variables. An algorithm has been developed in this research for achieving this goal.

Then, in a second phase, three types of objectives can be defined on the basis of the performance variables. The performance variables which are usually used to evaluate the system can be maximized, minimized, or kept to a target value. These objectives are then propagated in the causal graph and nodes, where contradictory objectives are detected and highlighted.

Those nodes constitute the nodes where contradictions are discovered and in the following step, which is not developed in this research, inventive methods such as the TRIZ or Synectic methods can be used to solve the contradictions.

This contribution provides practical automated support for detecting conflicts and contradictions that are not always easy to detect in technical systems.

# 4. Conclusion and Perspectives

The research work presented in this manuscript focused on two phases of development projects, modeling and simulation in the systems engineering in the early stages of the process project and planning.

## 4.1 Modeling and simulation at the early design stage

The main objective of this research work was to develop an approach to support modeling and simulation in the early stages of development. Those stages are characterized by a limitation on the knowledge and the nature of the information available. In this work the modeling and simulation aspects focused on the modeling and evaluation of the concepts of solutions generated at an early stage. This contribution is complementary to other work performed in our research group on the modeling aspects at requirements levels. The progress compared to the current state of the art in the area is, in the author's view, the following:

- the approach developed is generic and can be applied to a wide variety of multi-physics systems. In Publication V, the feasibility of the approach is demonstrated through the case of an air bearing. It can be imagined that the approach is generic enough to be applied to other fields of science too, such as economics and social science. Several publications have already demonstrated the feasibility of the concept in these domains;

- this research work has made it possible to extend existing causal ordering algorithms from the literature by classifying the system variables into different categories derived from bond graph theory and by considering the system model architecture more precisely;

- this approach has been implemented in the form of a computer software tool to support engineering designers in their decision making. However, the prototype software that was developed still requires improvements to reach a sufficient level of maturity to be commercialized.

The future work on this approach is that it first requires an ontology to be developed to automatically list system variables starting from the initial solution concepts. Second, the composability mechanism integrated into the approach in the form of a product theorem needs to be tested on a larger scale and tests need to be pursued in different fields. The last aspect that will require development and research is the extension of the system of units used in the methodology.

## 4.2 Project Planning

The proposed approach for project planning presented in this work has been developed to fulfill two goals:

- integrating a model-based approach with the development process architecture;

- improving the ease of use of existing scheduling approaches based on traditional genetic algorithms for practical use in an engineering context.

The first goal was to integrate requirements engineering and planning scheduling as early as possible in the development process. For this purpose requirement models were used as an initial basis for planning and scheduling. The requirements represent what needs to be developed in order to satisfy the stakeholders and complete the project. They can also form a good basis on which to define the structure of the project planning in the form of design tasks.

This way of deriving project activities and project tasks has the advantage of ensuring continuity between the definition of the requirement model and the process planning. From an engineering point of view it is also a valid standpoint. The author is nevertheless conscious that tasks and ac-

tivities can be derived from other viewpoints too, such as, for example, a generic development process viewpoint or an engineering speciality viewpoint.

This research work has demonstrated through classical literature case studies that the integration of a model-based approach for the requirements engineering associated with the model-based approach can be used for addressing these issues. The contributions generated by this research work are the following:

- the requirements are modeled using the SysML requirement diagram, which provides a set of predefined relationships in order to model and explore the dependencies with other requirements;

- SysML requirement diagrams are used for analyzing and communicating the requirements to the stakeholders in order to converge toward a common understanding;

- a set of tasks is considered as a model element in the systems model, which is related to the requirements. Therefore, this set of tasks can be automatically transferred into a process architecture DSM, which avoids the error humans are prone to when dealing with large-scale processes;

- the requirements and tasks in the system models can easily be reused; this can save time and costs when developing the systems in future projects.

In a large-scale project, several iterations/feedback cycles are usually introduced in the process architecture. Planning the project process also means organizing the tasks in order to minimize the iterations. Genetic Algorithms (GA) have been widely used to solve this problem.

However, as noted by the author, GA algorithms require many control parameters to be set. Therefore, this thesis has investigated the use of a Discrete Differential Evolution (DDE) algorithm to address the issue of the controlling parameters.

The proposed approach has performed testing of DDE algorithms on case studies used in other published research studies for comparison purposes (Publication II). As a result, the solutions obtained are very competitive in terms of their quality but are much easier to use than GA. In addition, the framework was applied to another case study of the development of a mobile robot and has been presented in Publication I too. This work demonstrated that it was possible to automatically reduce significantly the number of iterations required in the development process, the length of an iteration, and also to increase the concurrency of complex design developments using this approach.

The potential limitation of the approach is that it may face stagnation when there is no improvement in the solutions over the generations for some period of time. This is linked with the search for a global optimum. Such a type of approach cannot guarantee that the solution found is a global optimum in the search space.

For future research in this area, it is possible to investigate local searches for searching for a feasible solution in the local neighborhood in order to improve the quality of the solutions obtained. In addition, the impact of changes in requirements on the project scheduling needs to be investigated in future research.

# Bibliography


Bellagamba, L., *Systems Engineering and Architectting: Creating Formal Requirements*, CRC Press, Boca Raton, ISBN 978-1-4398-8140-8, 2012.

Bhaskar, R. and Nigam, A., *Qualitative physics using dimensional analysis*, Artif. Intell., vol. 45(1-2), pp. 73–111, ISSN 0004-3702, doi:10.1016/0004-3702(90)90038-2, URL http://dx.doi.org/10.1016/0004-3702(90)90038-2, Sep. 1990.

Boehm, B. W., *Software Engineering Economics*, Pearson Education, ISBN 0138221227, 1981.

Bridgman, P., *Dimensional Analysis*, Encyclopedia Britannica E.-i.-C. Wm. Haley, Editor, 1969.

Buckingham, E., *On Physically Similar Systems; Illustrations of the Use of Dimensional Analysis*, Physical Review, vol. 4(4), pp. 345–376, 1914.

Butterfield, R., *Dimensional Analysis Revisited*, n Institute of Mechanical Engineers, 2001.

Coatanéa, E., *Conceptual Modelling of Life Cycle Design - A Modelling and Evaluation Method Based on Analogies and Dimensionless Numbers*, Ph.D. thesis, Helsinki University of Technology and Université de Bretagne Occidentale, Espoo, Finland, October 2005.

Electronic Industry Association, *ANSI/EIA-632*, 1999.

Eppinger, S. D. and Browning, T. R., *Design Structure Matrix Methods and Applications*, The MIT Press, 1st edn., ISBN 9780262017527, 2012.

Estefan, J. A., *Survey of model-based systems engineering (MBSE) methodologies*, California Institute of Technology, Pasadena, California, USA May, vol. 25, 2007.

Fazar, W., *Program Evaluation and Review Technique*, The American Statistician, vol. 13(2), 1959.

Fernando, E. P. C., *Use of Interdependency Matrix for Expediting Implementation of an Integrated Development Programme in a Developing Country*, in *Proceedings of the Second International Congress for Project Planning by Network Analysis*, 1969.

Forbus, K. D., *Qualitative process theory*, Artifical Intelligence, vol. 24, pp. 85–168, 1984.

Forsberg, K. and Mooz, H., *Systems engineering for faster, cheaper, better*, in *In Proceedings of the ninth annual international symposium of the INCOSE*, Brighton, England, 1999.

Friedenthal, S., Moore, A. and Steiner, R., *A Practical Guide to SysML, 2nd Edition: The Systems Model Language*, Morgan Kaufmann, ISBN 9780123852069, 2011.

Gantt, H. L., *Work, Wages and Profit*, The Engineering Magazine, 1910.

Gebala, D. A. and Eppinger, S. D., *Methods for Analyzing Design Procedures*, in *Proceedings of the ASME Third International. Conference On Design Theory and Methodology*, Miami, F, September 1991.

Hayes, M., *The Role of Activity Precedence Relationships in Node-Oriented Networks*, in *Proceedings of the Second International Congress for Project Planning by Network Analysis*, 1969.

Holt, J., Perry, S., Brownsword, M., Cancila, D., Hallerstede, S. and Hansen, F., *Model-based requirements engineering for system of systems*, in *2012 7th International Conference on System of Systems Engineering (SoSE)*, pp. 561–566, doi:10.1109/SYSoSE.2012.6384145, 2012.

Hull, E., Jackson, K. and Dick, J., *Requirements Engineering, 2nd Edition*, Springer, London, 2nd edn., ISBN 1-85233-879-2, 2005.

IEEE-1, *IEEE Standard for Application and Management of the Systems Engineering Process*, IEEE Std 1220-2005 (Revision of IEEE Std 1220-1998), pp. 1–87, doi:10.1109/IEEESTD.2005.96469, 2005.

INCOSE, *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, International Council on Systems Engineering (INCOSE), San Diego, CA, USA, version 3.2.2, 2012.

de Jong, F. J., *Dimensional Analysis for Economists*, 1967.

Kalawsky, R., O'Brien, J., Chong, S., Wong, C., Jia, H., Pan, H. and Moore, P., *Bridging the Gaps in a Model-Based System Engineering Workflow by Encompassing Hardware-in-the-Loop Simulation*, Systems Journal, IEEE, vol. 7(4), pp. 593–605, ISSN 1932-8184, doi:10.1109/JSYST.2012.2230995, Dec 2013.

Kelley, J. and Walker, M., *Critical-Path Planning and Scheduling*, in *Proceedings of the Eastern Joint Computer Conference*, 1959.

Kokar, M., *Critical hypersurfaces and the quantity space*, in *Proceedings of AAAI-87*, 1987.

Kuipers, B., *Commonsense reasoning about causality: Deriving behavior from structure*, Artifical Intelligence, vol. 24(1-3), 1984.

Kusiak, A. and Wang, J., *Decomposition of the Design Process*, ASME J. Mech. Des., vol. 115(4), p. 687– 695, 1993.

McCulley, C. and Bloebaum, C. L., *A Genetic Tool for Optimal Design Sequencing in Complex Engineering Systems*, Structural optimization, vol. 12(2), pp. 186–201, October 1996.

Meier, C., Yassine, A. A. and Browning, T. R., *Design Process Sequencing With Competent Genetic Algorithms*, vol. 129(6), pp. 566–585, ISSN 1050-0472, doi: 10.1115/1.2717224, URL http://dx.doi.org/10.1115/1.2717224, 2006.

Nuseibeh, B. and Easterbrook, S., *Requirements Engineering: A Roadmap*, in *Proceedings of the Conference on The Future of Software Engineering*, ICSE '00, p. 35â46, ACM, ISBN 1-58113-253-0, doi:10.1145/336512.336523, URL http://doi.acm.org/10.1145/336512.336523, 2000.

OMG, *Unified Modeling Language (UML) specification version 2.4.1*, URL http://www.omg.org/spec/UML/2.4.1/Infrastructure/PDF, 2011.

OMG, *SysML specification version 1.3*, Object Management Group, URL http://www.omg.org/spec/SysML/1.3/PDF, June 2012.

Pyster, A. and Olwell, D. H., *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, URL www.sebokwiki.org/, 2013.

Ramon Costa, D. K. S. I., *Iteration in Engineering Design: Inherent and unavoidable or product of choices made?*, in *ASME International Design Engineering Technical Conferences (IDETC/CIE2003)*, pp. 669–674, Chicago, Illinois, 2003.

Ramos, A., Ferreira, J. and Barcelo, J., *Model-Based Systems Engineering: An Emerging Approach for Modern Systems*, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 42(1), pp. 101–111, ISSN 1094-6977, doi:10.1109/TSMCC.2011.2106495, 2012.

Schlager, J., *Systems engineering: key to modern development*, IRE Transactions, vol. EM-3, pp. 64–66, doi:10.1109/IRET-EM.1956.5007383, 1956.

Scott, J. A., *A strategy for Modeling the Design-Development Phase of a Product*, Ph.D. thesis, Department of Marine Engineering, University of Newcastle upon Tyne, Newcastle, UK, 1999.

Sheard, S. A., *Systems Engineering Standards and Models Compared*, in *Proceedings of the 8th International Symposium on Systems Engineering (INCOSE*, pp. 589–605, 1998.

Shen, Q. and Peng, T., *Combining Dimensional Analysis and Heuristics for Causal Ordering*, in *In Memory of Dr RobMilne – Rob Milne: A Tribute to a Pioneering AI Scientist Entrepreneur and Mountaineer, A. Bundy and S. Wilson (Eds.)*, pp. 39–53, IOS Press, 2006.

Soares, M. D. S. and Vrancken, J., *Model-Driven User Requirements Specification using SysML*, vol. 3(6), ISSN 1796-217X, doi:10.4304/jsw.3.6.57-68, URL http://ojs.academypublisher.com/index.php/jsw/article/view/1959, 2008.

Sonin, A., *The physical basis of dimensional analysis*, Department of Mechanical Engineering MIT Cambridge, MA 02139, 2001.

Stevens, R., Brook, P., Jackson, K. and Arnold, S., *System Engineering: copying with complexity*, Prentice Hall, ISBN 0-13-095085-8, 1998.

Steward, D., *The Design Structure System: A Method for Managing the Design of Complex Systems*, IEEE Transactions on Engineering Management, vol. 28(3), pp. 71–74, 1981.

Szirtes, T., *Applied Dimensional Analysis and Modeling*, Butterworth Heinemann, 2nd edn., ISBN 978-0-12-370620-1, 2006.

Tarjan, R., *Depth-First Search and Linear Graph Algorithms*, SIAM Journal on Computing, vol. 1(2), pp. 146–160, 1972.

Todd, D., *Multiple Criteria Genetic Algorithms in Engineering Design and Operation*, Ph.D. thesis, Engineering Design Centre, University of Newcastle upon Tyne, Newcastle, UK, 1997.

Ullman, D. G., *The mechanical design process*, McGraw-Hill, 3rd edn., ISBN 9780072373387, 2002.

Ullman, D. G., *The mechanical design process*, McGraw-Hill, 4th edn., ISBN 9781259002410, 2010.

Ulrich, K. T. and Eppinger, S. D., *Product Design and Development*, McGraw-Hill/Irwin, 5th edn., ISBN 978-0073404776, May 2011.

Warfield, J. N., *Binary Matrices in System Modeling*, IEEE Transactions on Systems, Man, and Cybernetics, vol. 3(5), pp. 441–449, 1973.

Yassine, A. and Braha, D., *Complex Concurrent Engineering and the Design Structure Matrix Method*, Concurrent Engineering, vol. 11(3), pp. 165–176, September 2003.

Yassine, A., Joglekar, N., Braha, D., Eppinger, S. and Whitney, D., *Information hiding in product development: the design churn effect*, Research in Engineering Design, vol. 14(3), pp. 145–161, November 2003.

Yassine, A., Whitney, D., Lavine and Zambito, T., *DO-IT-RIGHT-FIRST-TIME (DRFT) Approach to Design Structure Matrix (DSM) Restructuring*, in *International Conference on Design Theory and Methodology (DTM 2000)*, Baltimore, Maryland, USA, 2000.

Zave, P., *Classification of Research Efforts in Requirements Engineering*, ACM Comput. Surv., vol. 29(4), p. 315â321, ISSN 0360-0300, doi: 10.1145/267580.267581, URL `http://doi.acm.org/10.1145/267580.267581`, 1997.

Zeng, Y., *Recursive object model (ROM)—Modelling of linguistic information in engineering design*, Computers in Industry, vol. 59(6), pp. 612 – 625, ISSN 0166-3615, doi:http://dx.doi.org/10.1016/j.compind.2008.03.002, URL `http://www.sciencedirect.com/science/article/pii/S0166361508000249`, 2008.

Aalto-DD 19/2015

ISBN 978-952-60-6077-4 (printed)
ISBN 978-952-60-6078-1 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)

**Aalto University**
**School of Engineering**
**Department of Engineering Design and Production**
**www.aalto.fi**

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

**DOCTORAL DISSERTATIONS**